Generating Query Suggestions to Support Task-Based Search

Darío Garigliotti
University of Stavanger
dario.garigliotti@uis.no

Krisztian Balog
University of Stavanger
krisztian.balog@uis.no

ABSTRACT

We address the problem of generating query suggestions to support users in completing their underlying tasks (which motivated them to search in the first place). Given an initial query, these query suggestions should provide a coverage of possible subtasks the user might be looking for. We propose a probabilistic modeling framework that obtains keyphrases from multiple sources and generates query suggestions from these keyphrases. Using the test suites of the TREC Tasks track, we evaluate and analyze each component of our model.

CCS CONCEPTS

•**Information systems** →**Query suggestion;**

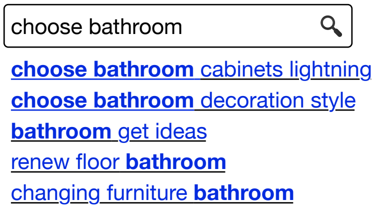
KEYWORDS

Query suggestions, task-based search, supporting search tasks

ACM Reference format:
Darío Garigliotti and Krisztian Balog. 2017. Generating Query Suggestions to Support Task-Based Search. In *Proceedings of SIGIR'17, August 7–11, 2017, Shinjuku, Tokyo, Japan,* 4 pages.
DOI: http://dx.doi.org/10.1145/3077136.3080745

1 INTRODUCTION

Search is often performed in the context of some larger underlying task [11]. There is a growing stream of research aimed at making search engines more task-aware (i.e., recognizing what task the user is trying to accomplish) and customizing the search experience accordingly (see §2). In this paper, we focus our attention on one particular tool for supporting task-based search: query suggestions. Query suggestions are an integral part of modern search engines [16]. We envisage an user interface where these suggestions are presented once the user has issued an initial query; see Figure 1. Note that this is different from query autocompletion, which tries to recommend various possible completions while the user is still typing the query. The task-aware query suggestions we propose are intended for exploring various aspects (subtasks) of the given task after inspecting the initial search results. Selecting them would allow the user to narrow down the scope of the search.



choose bathroom

choose bathroom cabinets lightning
choose bathroom decoration style
bathroom get ideas
renew floor **bathroom**
changing furniture **bathroom**

Figure 1: Query suggestions to support task-based search.

The Tasks track at the Text REtrieval Conference (TREC) has introduced an evaluation platform for this very problem, referred to as *task understanding* [20]. Specifically, given an initial query, the system should return a ranked list of keyphrases "that represent the set of all tasks a user who submitted the query may be looking for" [20]. The goal is to provide a complete coverage of subtasks for an initial query, while avoiding redundancy. We use these keyphrases as *query suggestions.*

Our aim is to generate such suggestions in a setting where past usage data and query logs are not available or cannot be utilized. This would be typical for systems that have a smaller user base (e.g., in the enterprise domain) or when a search engine has been newly deployed [4]. One possibility is to use query suggestion APIs, which are offered by all major web search engines. These are indeed one main source type we consider. Additionally, we use the initial query to search for relevant documents, using web search engines, and extract keyphrases from search snippets and from full text documents. Finally, given the task-based focus of our work, we lend special treatment to the WikiHow site,[1] which is an extensive database of how-to guides.

The main contribution of this paper is twofold. First, we propose a transparent architecture, using generative probabilistic modeling, for extracting keyphrases from a variety of sources and generating query suggestions from them. Second, we provide a detailed analysis of the components of our framework using different estimation methods. Many systems that participated in the TREC Tasks track have relied on strategic combinations of different sources to produce query suggestions, see, e.g., [7–9]. However, no systematic comparison of the different source types has been performed yet—we fill this gap. Additional components include estimating a document's importance within a given source, extracting keyphrases from documents, and forming query suggestions from these keyphrases. Finally, we check whether our findings are consistent across the 2015 and 2016 editions of the TREC Tasks track.

2 RELATED WORK

There is a large body of work on understanding and supporting users in carrying out complex search tasks. Log-based studies have been one main area of focus, including the identification of tasks and segmentation of search queries into tasks [2, 14] and mining task-based search sessions in order to understand query

SIGIR'17, August 7–11, 2017, Shinjuku, Tokyo, Japan
© 2017 ACM. 978-1-4503-5022-8/17/08...$15.00
DOI: http://dx.doi.org/10.1145/3077136.3080745

[1] http://www.wikihow.com/

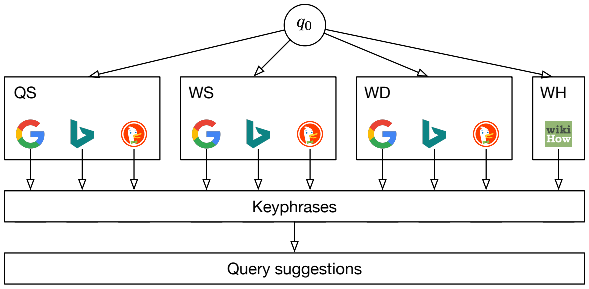


Figure 2: High-level overview of our approach.

reformulations [10] or search trails [19]. Another theme is supporting exploratory search, where users pursue an information goal to learn or discover more about a given topic. Recent research in this area has brought the importance of support interfaces into focus [1, 3, 17]. Our main interest is in query suggestions, as a distinguished support mechanism. Most of the related work utilizes large-scale query logs. For example, Craswell and Szummer [6] perform a random walk on a query-click graph. Boldi et al. [5] model the query flow in user search sessions via chains of queries. Scenarios in the absence of query logs have been addressed in [4, 13], where query suggestions are extracted from the document corpus. However, their focus is on query autocompletion, representing the completed and partial terms in a query. Kelly et al. [12] have shown that users prefer query suggestions, rather than term suggestions. We undertake the task of suggesting queries to users, related to the task they are performing, as we shall explain in the next section.

3 PROBLEM STATEMENT

We adhere to the problem definition of the task understanding task of the TREC Tasks track. Given an initial query q_0, the goal is to return a ranked list of query suggestions $\langle q_1, \dots q_n \rangle$ that cover all the possible subtasks related to the task the user is trying to achieve. In addition to the initial query string, the entities mentioned in it are also made available (identified by their Freebase IDs).

For example, for the query "low wedding budget," subtasks include (but are not limited to) "buy a used wedding gown," "cheap wedding cake," and "make your own invitations." These subtasks have been manually identified by the track organizers based on information extracted from the logs of a commercial search engine. The suggested keyphrases are judged with respect to each subtask on a three point scale (non-relevant, relevant, and highly relevant). Note that subtasks are only used in the evaluation, these are not available when generating the keyphrases.

4 APPROACH

We now present our approach for generating query suggestions. As Figure 2 illustrates, we obtain keyphrases from a variety of sources, and then construct a ranked list of query suggestions from these.

4.1 Generative Modeling Framework

We introduce a generative probabilistic model for scoring the candidate query suggestions according to $P(q|q_0)$, i.e., the probability that a query suggestion q was generated by the initial query q_0. Formally:

$$\begin{aligned} P(q|q_0) &= \sum_s P(q|q_0, s)P(s|q_0) \\ &= \sum_s \Big(\sum_d P(q|q_0, s, d)P(d|q_0, s)\Big) P(s|q_0) \\ &= \sum_s \Big(\sum_d \Big(\sum_k P(q|q_0, s, k)P(k|s, d)\Big) P(d|q_0, s)\Big) P(s|q_0)\ . \end{aligned}$$

This model has four components: (i) $P(s|q_0)$ expresses the importance of a particular information source s for the initial query q_0; (ii) $P(d|q_0, s)$ represents the importance of a document d originating from source s, with respect to the initial query; (iii) $P(k|d, s)$ is the relevance of a keyphrase k extracted from a document d from source s; and (iv) $P(q|q_0, s, k)$ is the probability of generating query suggestion q, given keyphrase k, source s, and the initial query q_0. Below, we detail the estimation of each of these components.

4.2 Source Importance

We collect relevant information from four kinds of sources: query suggestions (QS), web search snippets (WS), web search documents (WD), and WikiHow (WH). For the first three source types, we use three different web search engines (Google, Bing, and DuckDuckGo), thereby having a total of 10 individual sources. In this work, we assume conditional independence between a source s and the initial query q_0, i.e., set $P(s|q0) = P(s)$.

4.3 Document Importance

From each source s, we obtain the top-K ($K = 10$) documents for the query q_0. We propose two ways of modeling the importance of a document d originating from s: (i) uniform and (ii) inversely proportional to the rank of d among the top-K documents, that is:

$$P(d|q_0, s) = \frac{K - r + 1}{\sum_{i=1}^{K} K - i + 1} = \frac{K - r + 1}{K(K + 1)/2}\ ,$$

where r is the rank position of d ($r \in [1..K]$).

4.4 Keyphrase Relevance

We obtain keyphrases from each document, using an automatic keyphrase extraction algorithm. Specifically, we use the RAKE keyword extraction system [15]. For each keyphrase k, extracted from document d, the associated confidence score is denoted by $c(k, d)$. Upon a manual inspection of the extraction output, we introduce some data cleansing steps. We only retain keyphrases that: (i) have an extraction confidence above an empirically set threshold of 2; (ii) are at most 5 terms long; (iii) each of the terms has a length between 4 and 15 characters, and is either a meaningful number (i.e., max. 4 digits) or a term (excluding noisy substrings and reserved keywords from mark-up languages). Finally, we set the relevance of k as $P(k|d, s) = c(k, d)/\sum_{k'} c(k', d)$.

In case s is of type QS, each returned document actually corresponds to a query suggestion. Thus, we treat each of these documents d as a single keyphrase k, for which we set $P(k|d, s) = 1$.

Table 1: Comparison of query suggestion generators across the different types of sources. Statistical significance is tested against the corresponding line in the top block.

$P(q\|q_0, s, k)$	S	2015		2016	
		ERR-IA	α-NDCG	ERR-IA	α-NDCG
Using raw keyphrases	QS	0.0755	0.1186	**0.4114**	**0.5289**
	WS	**0.2011**	**0.2426**	0.3492	0.4038
	WD	0.1716	0.2154	0.2339	0.2886
	WH	0.0744	0.1044	0.1377	0.1723
Using expanded keyphrases	QS	0.0751	0.1182	**0.4046**	**0.5233**
	WS	**0.1901**	**0.2274**	$0.2927^{\dagger}$	$0.3467^{\dagger}$
	WD	0.1551	0.2097	$0.1045^{\ddagger}$	$0.1667^{\ddagger}$
	WH	0.0849	0.1090	$0.0789^{\dagger}$	$0.0932^{\ddagger}$

Table 2: Comparison of document importance estimators across the different types of sources. Statistical significance is tested against the corresponding line in the top block.

$P(d\|q_0, s)$	S	2015		2016	
		ERR-IA	α-NDCG	ERR-IA	α-NDCG
Uniform	QS	0.0755	0.1186	**0.4114**	**0.5289**
	WS	**0.2011**	**0.2426**	0.3492	0.4038
	WD	0.1716	0.2154	0.2339	0.2886
	WH	0.0849	0.1090	0.1377	0.1723
Rank-based decay	QS	$0.0891^{\dagger}$	$0.1307^{\dagger}$	**0.4288**	**0.5455**
	WS	**0.1906**	**0.2315**	0.3386	0.4011
	WD	0.1688	0.2119	0.1964	0.2608
	WH	0.0935	0.1225	0.1195	0.1495

4.5 Query Suggestions

As a final step, we need to generate query suggestions from the extracted keyphrases. As a baseline option, we take each raw keyphrase k as-is, i.e., with $q = k$ we set $P(q|q_0, s, k) = 1$.

Alternatively, we can form query suggestions by *expanding keyphrases*. Here, k is combined with the initial query q_0 using a set of expansion rules proposed in [7]: (i) adding k as a suffix to q_0; (ii) adding k as a suffix to an entity mentioned in q_0; and (iii) using k as-is. Rules (i) and (ii) further involve a custom string concatenation operator; we refer to [7] for details. Each query suggestion q, that is generated from keyword k, has an associated confidence score $c(q, q_0, s, k)$. We then set $P(q|q_0, s, k) = c(q, q_0, s, k)/ \sum_{q'} c(q', q_0, s, k)$. By conditioning the suggestion probability on s, it is possible to apply a different approach for each source. Like in the previous subsection, we treat sources of type QS distinctly, by simply taking $q = k$ and setting $P(q|q_0, s, k) = 1$.

We note that it is possible that multiple query suggestions have the same final probability $P(q|q_0)$. We resolve ties using a deterministic algorithm, which orders query suggestions by length (favoring short queries) and then sorts them alphabetically.

5 RESULTS

In this section we present our experimental setup and results.

5.1 Experimental Setup

We use the test suites of the TREC 2015 and 2016 Tasks track [18, 20]. These contain 34 and 50 queries with relevance judgments, respectively. We report on the official evaluation metrics used at the TREC Tasks track, which are ERR-IA@20 and α-NDCG@20. In accordance with the track's settings, we use ERR-IA@20 as our primary metric. (For simplicity, we omit mentioning the cut-off rank of 20 in all the table headers.) We noticed that in the ground truth the initial query itself has been judged as a highly relevant suggestion in numerous cases. We removed these cases, as they make little sense for the envisioned scenario; we note that this leads to a drop in absolute terms of performance. We report on statistical significance using a two-tailed paired t-test at $p < 0.05$ and $p < 0.001$, denoted by † and ‡, respectively.

In a series of experiments, we evaluate each component of our approach, in a bottom-up fashion. For each query set, we pick the configuration that performed best on that query set, which is an idealized scenario. Note that our focus is not on absolute performance figures, but on answering the following research questions:

- **RQ1** What are the most useful information sources?
- **RQ2** What are effective ways of (i) estimating the importance of documents and (ii) generating query suggestions from keyphrases?
- **RQ3** Are our findings consistent across the two query sets?

5.2 Query Suggestion Generation

We start our experiments by focusing on the generation of query suggestions and compare the two methods described in §4.5. The document importance is set to be uniform. We report performance separately for each of the four source types S (that is, we set $P(s)$ uniformly among sources $s \in S$ and set $P(s) = 0$ for $s \notin S$). Table 1 presents the results. It is clear that, with a single exception (2015 WH), it is better to use the raw keyphrases, without any expansion. The differences are significant on the 2016 query set for all source types but QS. Regarding the comparison of different source types, we find that QS > WS > WD > WH on the 2016 query set, meanwhile for 2015, the order is WS > WD > QS, WH.

5.3 Document Importance

Next, we compare the two document importance estimator methods, uniform and rank-based decay (cf. §4.3), for each source type. Table 2 reports the results. We find that rank-based document importance is beneficial for the query suggestion (QS) source types, for both years, and for WikiHow (WH) on the 2015 topics. However, the differences are only significant for QS 2015. For all other source types, the uniform setting performs better.

We also compare performance across the 10 individual sources. Figure 3 shows the results, in terms of ERR-IA@20, using the uniform estimator. We observe a very similar pattern using the rank-based estimator (which is not included due to space constraints). On the 2016 query set, the individual sources follow the exact same patterns as their respective types (i.e., QS > WS > WD > WH), with one exception. The Bing API returned an empty set of search suggestions for many queries, hence the low performance of QS_{Bing}. We can observe a similar pattern on the 2015 topics, with the exception of sources of type QS, which are the least effective here.

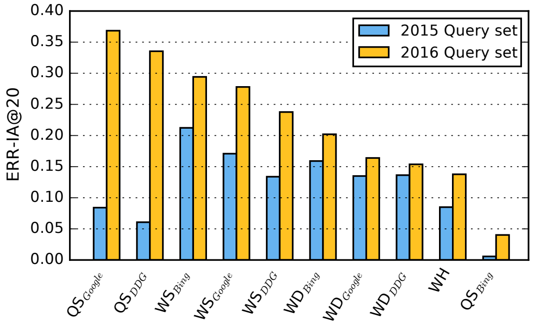


Figure 3: Performance of individual sources, sorted by performance on the 2016 query set.

5.4 Source Importance

Finally, we combine query suggestions across different sources; for that, we need to set the importance of each source. We consider three different strategies for setting $P(s)$: (i) uniformly; (ii) proportional to the importance of the corresponding source type (QS, WS, WD, and WH) from the previous step (cf. Table 2); (iii) proportional to the importance of the individual source (cf. Figure 3). The results are presented in Table 3. Firstly, we observe that the combination of sources performs better than any individual source type on its own. As for setting source importance, on the 2015 query set we find that (iii) delivers the best results, which is in line with our expectations. On the 2016 query set, only minor differences are observed between the three methods, none of which are significant.

5.5 Summary of Findings

(RQ1) Query suggestions provided by major web search engines are unequivocally the most useful information source on the 2016 queries. We presume that these search engine suggestions are already diversified, which we can directly benefit from for our task. These are followed, in order, by keyphrases extracted from (i) web search snippets, (ii) web search results, i.e., full documents, and (iii) WikiHow articles. On the 2015 query set, query suggestions proved much less effective; see RQ3 below. (RQ2) With a single exception, using the raw keyphrases, as-is, performs better than expanding them by taking the original query into account. For web query suggestions it is beneficial to consider the rank order of suggestions, while for web search snippets and documents the uniform setting performs better. For WikiHow, it varies across query sets. (RQ3) Our main observations are consistent across the 2015 and 2016 query sets, regarding documents importance estimation and suggestions generation methods. It is worth noting that some of our methods were officially submitted to TREC 2016 [7] and were included in the assessment pools. This is not the case for 2015, where many of our query suggestions are missing relevance assessments (and, thus, are considered irrelevant). This might explain the low performance of QS sources on the 2015 queries.

6 CONCLUSIONS

In this paper, we have addressed the task of generating query suggestions that can assist users in completing their tasks. We have proposed a probabilistic generative framework with four components: source importance, document importance, keyphrase relevance, and query suggestions. We have proposed and experimentally compared various alternatives for these components.

One important element, missing from our current model, is the representation of specific subtasks. As a next step, we plan to cluster query suggestions together that belong to the same subtask. This would naturally enable us to provide diversified query suggestions.

Table 3: Combination of all sources using different source importance estimators. Significance is tested against the uniform setting (line 1).

$P(s)$	2015		2016	
	ERR-IA	α-NDCG	ERR-IA	α-NDCG
Uniform	0.2219	0.2835	0.4561	0.5793
Source-type	0.2381	0.2905	**0.4570**	**0.5832**
Individual	**0.2518**†	**0.3064**†	0.4562	**0.5832**

REFERENCES

[1] Salvatore Andolina, Khalil Klouche, Jaakko Peltonen, Mohammad E. Hoque, Tuukka Ruotsalo, Diogo Cabral, Arto Klami, Dorota Glowacka, Patrik Floréen, and Giulio Jacucci. 2015. IntentStreams: Smart Parallel Search Streams for Branching Exploratory Search. In *Proc. of IUI*. 300–305.

[2] Ahmed H. Awadallah, Ryen W. White, Patrick Pantel, Susan T. Dumais, and Yi-Min Wang. 2014. Supporting Complex Search Tasks. In *Proc. of CIKM*. 829–838.

[3] Krisztian Balog. 2015. Task-completion Engines: A Vision with a Plan. In *Proc. of the 1st International Workshop on Supporting Complex Search Tasks*.

[4] Sumit Bhatia, Debapriyo Majumdar, and Prasenjit Mitra. 2011. Query Suggestions in the Absence of Query Logs. In *Proc. of SIGIR*. 795–804.

[5] Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, Aristides Gionis, and Sebastiano Vigna. 2008. The Query-flow Graph: Model and Applications. In *Proc. of CIKM*. 609–618.

[6] Nick Craswell and Martin Szummer. 2007. Random walks on the click graph. In *Proc. of SIGIR*. 239–246.

[7] Darío Garigliotti and Krisztian Balog. 2016. The University of Stavanger at the TREC 2016 Tasks Track. In *Proc. of TREC*.

[8] Matthias Hagen, Steve Göring, Magdalena Keil, Olaoluwa Anifowose, Amir Othman, and Benno Stein. 2015. Webis at TREC 2015: Tasks and Total Recall Tracks. In *Proc. of TREC*.

[9] Matthias Hagen, Johannes Kiesel, Payam Adineh, Masoud Alahyari, Ehsan Fatehifar, Arafeh Bahrami, Pia Fichtl, and Benno Stein. 2016. Webis at TREC 2016: Tasks, Total Recall, and Open Search Tracks. In *Proc. of TREC*.

[10] Jiepu Jiang, Daqing He, Shuguang Han, Zhen Yue, and Chaoqun Ni. 2012. Contextual Evaluation of Query Reformulations in a Search Session by User Simulation. In *Proc. of CIKM*. 2635–2638.

[11] Diane Kelly, Jaime Arguello, and Robert Capra. 2013. NSF Workshop on Task-based Information Search Systems. *SIGIR Forum* 47, 2 (2013), 116–127.

[12] Diane Kelly, Karl Gyllstrom, and Earl W. Bailey. 2009. A Comparison of Query and Term Suggestion Features for Interactive Searching. In *Proc. of SIGIR*. 371–378.

[13] Udo Kruschwitz, Deirdre Lungley, M-Dyaa Albakour, and Dawei Song. 2013. Deriving query suggestions for site search. *JASIST* 64, 10 (2013), 1975–1994.

[14] Claudio Lucchese, Salvatore Orlando, Raffaele Perego, Fabrizio Silvestri, and Gabriele Tolomei. 2013. Discovering Tasks from Search Engine Query Logs. *ACM Trans. Inf. Syst.* 31, 3, Article 14 (2013), 43 pages.

[15] Alyona Medelyan. 2015. Modified RAKE algorithm. https://github.com/zelandiya/RAKE-tutorial. (2015). Accessed: 2017-01-23.

[16] Umut Ozertem, Olivier Chapelle, Pinar Donmez, and Emre Velipasaoglu. 2012. Learning to Suggest: A Machine Learning Framework for Ranking Query Suggestions. In *Proc. of SIGIR*. 25–34.

[17] Tuan A. Tran, Sven Schwarz, Claudia Niederée, Heiko Maus, and Nattiya Kanhabua. 2016. The Forgotten Needle in My Collections: Task-Aware Ranking of Documents in Semantic Information Space. In *Proc. of CHIIR*. 13–22.

[18] Manisha Verma, Evangelos Kanoulas, Emine Yilmaz, Rishabh Mehrotra, Ben Carterette, Nick Craswell, and Peter Bailey. 2016. Overview of the TREC Tasks Track 2016. In *Proc. of TREC*.

[19] Ryen W. White and Jeff Huang. 2010. Assessing the Scenic Route: Measuring the Value of Search Trails in Web Logs. In *Proc. of SIGIR*. 587–594.

[20] Emine Yilmaz, Manisha Verma, Rishabh Mehrotra, Evangelos Kanoulas, Ben Carterette, and Nick Craswell. 2015. Overview of the TREC 2015 Tasks Track. In *Proc. of TREC*.