Information Retrieval Journal manuscript No.
(will be inserted by the editor)

Learning to Rank Query Suggestions for Adhoc and Diversity Search

Rodrygo L. T. Santos · Craig Macdonald · Iadh Ounis

Received: date / Accepted: date

Abstract Query suggestions have become pervasive in modern web search, as a mechanism to guide users towards a better representation of their information need. In this article, we propose a ranking approach for producing effective query suggestions. In particular, we devise a structured representation of candidate suggestions mined from a query log that leverages evidence from other queries with a common session or a common click. This enriched representation not only helps overcome data sparsity for long-tail queries, but also leads to multiple ranking criteria, which we integrate as features for learning to rank query suggestions. To validate our approach, we build upon existing efforts for web search evaluation and propose a novel framework for the quantitative assessment of query suggestion effectiveness. Thorough experiments using publicly available data from the TREC Web track show that our approach provides effective suggestions for adhoc and diversity search.

1 Introduction

Web search queries are typically short, ill-defined representations of more complex information needs [26]. As a result, they can lead to unsatisfactory retrieval performance. Query suggestions have been introduced as a mechanism to attenuate this problem. Such a mechanism builds upon the vast amount of querying behaviour recorded by search engines in the form of query logs, in order to suggest related queries previously issued by other users with a similar information need [42]. The mined suggestions can be exploited in a variety of ways. For instance, a suggestion identified with high confidence can be considered for automatically rewriting the user's initial query [28]. Alternatively,

Rodrygo L. T. Santos · Craig Macdonald · Iadh Ounis
School of Computing Science, University of Glasgow, Glasgow, G12 8QQ, UK
Tel.: +44-141-330-6292
Fax: +44-141-330-4913
E-mail: {rodrygo,craigm,ounis}@dcs.gla.ac.uk

a few high quality suggestions can be offered to the user as alternatives to the initial query [4], or to help diversify the results retrieved for this query [38].

Several approaches have been proposed in recent years to mine query suggestions from a query log. These exploit a variety of available evidence to estimate the relevance of a candidate suggestion for a given query, including whether the query and the suggestion share common terms [4], common user sessions [23], or common clicked results [6,19]. Despite their relative success, most of these approaches share a common shortcoming. In particular, they underperform and can even fail to produce any relevant suggestion for queries with sparse or no past usage, which amount to a substantial fraction of the web search traffic [22]. To combat data sparsity, we propose to tackle query suggestion as a ranking problem. To this end, we devise an enriched representation of candidate suggestions in the space of the unique *terms* (as opposed to entire queries) present in a query log. In particular, we differentiate between all the terms related to a candidate suggestion according to whether they occur in the suggestion itself, or in other queries that share a session or a clicked document with the suggestion. This enriched representation leads to multiple criteria for ranking suggestions with respect to a query, which we integrate as query-dependent features in a unified ranking model automatically learned from training data. To further improve this model, we propose several query-independent features as quality indicators for a candidate suggestion.

In order to validate our approach, we propose a framework for quantitatively assessing query suggestion effectiveness. This framework departs from the often unreliable user assessment of the relevance of a suggestion [25] and measures the *actual* relevance of each suggestion when used as a replacement for the initial query in a reference adhoc retrieval system. Alternatively, as a user query can be ambiguous [43], our framework also enables the assessment of the diversity of a set of suggestions, in terms of how useful these suggestions are when used by a reference diversification system. To instantiate this framework in our evaluation, we use a web search engine as the reference adhoc retrieval system and a state-of-the-art diversification approach as the reference diversification system [38]. Moreover, we leverage standard document relevance assessments from existing web search evaluation campaigns for adhoc and diversity search. Our analysis shows that our suggestion evaluation framework is robust to missing relevance assessments from the underlying test collections. In particular, we evaluate our proposed approach at providing effective suggestions for 148 queries from the TREC 2009, 2010, and 2011 Web tracks [13,15,16]. The results of this investigation show that our learning to rank approach improves upon a state-of-the-art query suggestion baseline [7], while comparing favourably to the suggestion mechanism of a commercial web search engine, which arguably has access to more abundant data. Moreover, our produced suggestions are effective for both adhoc and diversity search, even when the original query is not present in the query log.

The major contributions of this article are:

1. We propose a framework for the quantitative evaluation of query suggestion effectiveness, building upon existing efforts for web search evaluation.
2. We validate our evaluation framework in terms of its robustness to missing relevance assessments from the underlying web test collections.
3. We propose a learning to rank approach for the query suggestion problem, leveraging multiple ranking features mined from a structured representation of candidate query suggestions.
4. We thoroughly evaluate our proposed approach using a publicly available web test collection for adhoc and diversity search.

The remainder of this article is organised as follows. Section 2 discusses related approaches for the query suggestion problem. Section 3 describes our approach for learning to rank query suggestions. Section 4 formalises our evaluation methodology. Sections 5 and 6 describe the experimental setup and the results of our evaluation. Finally, Section 7 presents our concluding remarks.

2 Related Work

Query suggestion has attracted considerable interest from the information retrieval community in recent years, particularly in the setting of query log mining [42]. Most query suggestion mechanisms attempt to infer the relevance of a candidate suggestion for a given query based on their textual similarity, their co-occurrence in common sessions, or their common clicked URLs. For instance, Jones et al. [28] proposed to generate candidate suggestions from co-session queries with a common substring. The strength of the relationship between the query and each candidate suggestion was further estimated by leveraging various similarity features, such as the edit distance and the mutual information between these queries. Analogously, Wang and Zhai [47] proposed to mine term association patterns from a query log. Their approach analysed the co-occurrence of terms in multi-word co-session queries and built a translation model for mining query suggestions.

A session-based approach was proposed by Fonseca et al. [23]. In particular, they deployed an association rule mining algorithm in order to identify query pairs with sufficient co-occurrence across multiple sessions. Such rules were then used as the basis for identifying query suggestions from a query log. Relatedly, Zhang and Nasraoui [50] exploited the sequence of queries in a query log session. Their approach created a graph with edges between consecutive queries in each session, weighted by these queries' textual similarity. A candidate suggestion for a given query was then scored based on the length of the path between the two queries, accumulated across all sessions in a query log where the query and the suggestion co-occurred.

A click-based approach was proposed by Baeza-Yates et al. [4]. In particular, they proposed to cluster queries represented using the terms present in the URLs clicked for these queries. Given an input query, candidate suggestions

from the same cluster as the query were then weighted based on their similarity to the query and their success rate, as measured by their fraction of clicked results in a query log. Relatedly, Mei et al. [30] exploited random walks on a bipartite query-click graph. In particular, they weighted a candidate suggestion for a query based on its 'hitting' time (i.e., the time it took for the node representing this query suggestion to be visited for the first time) for a random walk starting from the input query. Similarly, Boldi et al. [6] proposed to weight candidate suggestions by performing a short random walk on different slices of a query-flow graph, a query transition graph with edges classified as generalisations, specialisations, error corrections, or parallel moves [5].

Random walk approaches are generally regarded as the state-of-the-art for the query suggestion problem [42]. On the other hand, these approaches may suffer from data sparsity when providing suggestions for rare or unseen queries [22]. To overcome this issue, Szpektor et al. [46] proposed the notion of query template, a generalisation of a query in which entities are replaced with their type. By enriching the query-flow graph [5] with query templates, their approach was able to effectively generate suggestions for long-tail queries. A different approach aimed at tackling query sparsity was proposed by Broccolo et al. [7]. In particular, they proposed to index each query in a query log as a *virtual document* comprising the terms in the query itself and those of other queries from common sessions. As a result, they cast the query suggestion problem as a standard search over the inverted index of virtual documents.

In Section 3, we build upon the query representation strategy proposed by Broccolo et al. [7], which was shown to perform at least as effectively as the state-of-the-art query-flow graph approach of Boldi et al. [6] for head queries, while consistently outperforming it for queries with little or no past evidence [7, Section 4.4]. Inspired by this approach, we devise a *structured* virtual document representation by treating terms from different sources as distinct *fields*. In particular, besides the candidate suggestion itself and its co-session queries, we also leverage evidence from queries that share at least one click with the suggestion. These multiple sources of evidence are then used to produce multiple features for learning to rank query suggestions.

In this vein, Dang et al. [20] proposed a learning approach to identify effective terms from a query log to be appended to an input query. More recently, Song et al. [44] proposed a learning approach to produce diverse suggestions in response to a query. While also employing learning to rank, our approach differs from the aforementioned approaches in two fundamental ways. In particular, while Dang et al. [20] identified effective *expansion terms*, we are interested in the more general problem of query suggestion. As for the approach of Song et al. [44], instead of relying on the human assessment of suggestion effectiveness, which can be misleading [25], we explicitly incorporate the *observed* retrieval effectiveness (in terms of adhoc and diversity search) of a set of candidate suggestions in order to guide the learning process.

Lastly, query suggestions have been recently used to improve the diversity of web search results. In particular, Santos et al. [38] proposed to exploit query suggestions within a probabilistic framework for search result diversification.

Their xQuAD framework promoted search results that satisfied multiple query 'aspects' (represented as different query suggestions), provided that these aspects were not already well satisfied by the other results in the ranking. In Section 4, we employ this framework as the reference diversification system for evaluating our suggestion ranking mechanism. As the top performing approach at the diversity task of the TREC 2009, 2010, and 2011 Web tracks [13, 15, 16], it is a representative of state-of-the-art diversification mechanisms.

3 Learning to Rank Query Suggestions

With the abundant usage data available to commercial web search engines, query suggestion has traditionally been approached as a data-driven problem, as exemplified by the various approaches described in Section 2. While different approaches have exploited such rich data with more or less success, we argue that the relevance of a suggestion with respect to a query cannot be fully explained by one single criterion. Instead, we propose to estimate this relevance by leveraging multiple ranking criteria within a supervised learning to rank setting. As a result, not only do we move beyond the traditional approaches to query suggestion, but we make it possible to leverage these otherwise successful approaches as additional features in a robust query suggestion model.

In the following, Section 3.1 discusses alternative criteria for producing an initial sample of candidate suggestions to be used by our learning to rank approach. Section 3.2 describes our learning approach deployed to rerank the produced samples. Lastly, Section 3.3 describes our proposed features to represent a candidate suggestion for ranking.

3.1 Sampling Query Suggestions

Typically, learning to rank approaches operate on top of a *sample* of results retrieved by a standard ranking model [29], such as BM25 [37]. The results in this sample are then used by the learning approach to produce a feature-rich ranking model, which will be later used to rank the results retrieved for unseen queries. An effective sample should have high recall, so as to increase the number of relevant training examples available for learning to rank [29]. In the case of query suggestions, such a sample can be obtained by exploiting the rich information about a user query contained in a query log. In particular, we can describe a query log $\mathcal{L}$ as a set of records $\langle q_i, u_i, t_i, \mathcal{V}_i, \mathcal{C}_i \rangle$, where q_i is a query issued by user u_i at timestamp t_i. For this query, the user was shown a set of results $\mathcal{V}_i$ and clicked on the subset $\mathcal{C}_i \subseteq \mathcal{V}_i$. Typically, queries issued by the same user within a short timeframe (say, 30 min) are further grouped into a logical session, ideally reflecting a cohesive search mission [42].

As discussed in Section 2, most query suggestion approaches in the literature exploit the co-occurrence of queries in a session or their clicks in a common search result in order to produce effective suggestions. However, these

approaches underperform for rare or unseen queries [42]. In the former case, there is little evidence of the query's co-occurrence with potential suggestions in the query log. In the latter case, the initial query itself cannot even be located in the log. To tackle this sparsity problem, Broccolo et al. [7] proposed to represent queries in a query log as *virtual documents*. This bag-of-words representation comprises not only the words in the query itself, but also those present in other queries with a common session in the log. Such a representation combats data sparsity, since even previously unseen queries (i.e., queries without an exact match in the query log) will likely have at least one of their constituent words present in the log, which in turn may occur frequently in the virtual document representation of a relevant suggestion. Additionally, this representation enables the suggestion problem to be efficiently tackled as a standard search over an inverted index, with the potential to scale to extremely large query logs [21]. On the other hand, this representation lacks a more fine-grained treatment of the multiple evidence available for ranking, by not distinguishing between words from different sources.

To address this issue and to produce an effective sample of candidate suggestions to be used for learning to rank, we improve upon this bag-of-words representation by considering each available source of evidence as a separate *field* in a *structured virtual document*.[1] As a result, words that appear in a query suggestion can be weighted differently from those that appear in related queries with a common session. Moreover, we integrate an additional source of evidence as a third field in our structured virtual document representation. In particular, for each candidate suggestion, we also store words from queries with at least one common click in the query log. As an illustrative example, Figure 1 shows an excerpt of the structured virtual document representing 'metallica' as a candidate suggestion, highlighting this query itself (Q), co-session queries (S), and queries with a common click (C) as separate fields. Also note the 'count' attribute for each entry (E) in Figure 1, which denotes the frequency with which this entry co-occurs with 'metallica' in the entire query log (e.g., the queries 'metallica' and 'james hetfield' have 60 common clicks). During indexing, the term frequency of each entry is adjusted according to this value, so that the strength of the relationship between the entry and the candidate suggestion it relates to is captured appropriately.

When retrieving a sample of suggestions for a given query, there are multiple choices regarding which of the available fields to use: different choices lead to different samples for the same query (e.g., a sample of suggestions built by searching the Q field will probably be different from a sample based upon the S or C fields). A more fundamental question is which queries from the query log should be considered as candidate suggestions for any given query. In their approach, Broccolo et al. [7] used only queries that ended a *satisfactory* session (i.e., a session with at least one click in the last query). Arguably, non-satisfactory sessions (i.e., sessions with no clicks or without clicks on the

[1] An analogy to the document ranking problem can be made in which field-based models, such as BM25F [48], leverage evidence from fields such as the title, body, URL, or the anchor text of incoming hyperlinks in order to score a document.

```
<DOC>
  <DOCNO> metallica </DOCNO>
  <Q> metallica </Q>
  <S> <E count="1"> metalica </E>
      <E count="1"> queensryche </E>
      <E count="1"> ac dc </E>
      <E count="1"> pantera </E>
        ... </S>
  <C> <E count="4"> history of mettalica </E>
      <E count="1"> metallica concerts </E>
      <E count="18"> metclub </E>
      <E count="60"> james hetfield </E>
        ... </C>
</DOC>
```

Fig. 1 Example structured virtual document representation of the suggestion 'metallica'.

last query in the session) can also contribute potentially relevant suggestions. Moreover, non-final queries in both satisfactory and non-satisfactory sessions may also be useful. In Section 6, we will investigate multiple structured virtual document representations based on different combinations of the available fields (i.e., Q, S, and C), as well as different sampling criteria (i.e., whether to index queries from all sessions or from only satisfactory sessions, and whether to index all or only the last query in each of these sessions).

A breakdown of these alternative representations in terms of the storage overhead incurred by each of them is provided in Table 1. Firstly, restricting the index to comprise only satisfactory sessions or only the last query in each session naturally reduces the required storage space, since fewer queries are considered as candidate suggestions. More interestingly, compared to a suggestion representation based upon the query string (Q) only, a representation enriched with co-session (S) and co-clicked (C) queries does not affect the asymptotic space complexity of our approach. Indeed, all increases in space requirements stay within an order of magnitude of the space required to store the query alone. In particular, the most space-consuming representation (QSC) requires only 6.7 times more storage space (4.4 times after compression with gzip[2]) compared to the least space-consuming one (Q).

3.2 Learning a Query Suggestion Model

Given a query q and an indexed query log $\mathcal{L}$, we can now define query suggestion as the problem of retrieving a list of queries $\mathcal{Q}(q) \subseteq \mathcal{L}$ that could serve as effective alternatives to q. In particular, such alternatives could help the user better specify the information need originally expressed by q, or to diversify the search results produced for this query, in the hope of providing the user with at least one relevant result in the final ranking.

To tackle query suggestion as a learning to rank problem, we must learn an optimal ranking function $h : \mathcal{X} \rightarrow \mathcal{Y}$, mapping the input space $\mathcal{X}$ to the output space $\mathcal{Y}$. In particular, we define the input space $\mathcal{X}$ of a query q as

[2] http://www.gnu.org/software/gzip

Table 1 Space requirements for storing each of the seven considered structured virtual document representations: Q, S, C, QS, QC, SC, QSC. The top half of the table shows the total uncompressed size (in MB) of the resulting corpora of virtual documents, whereas the bottom half shows the size of each corpus after compression. In both cases, the percentage figures denote the incurred overhead compared to storing only the query string (Q) of each suggestion. The total number of suggestions in each corpus is shown in the bottom row.

	sessions	all				satisfactory			
	queries	all		last		all		last	
uncompressed	Q	141.7		78.3		86.4		44.2	
	S	513.4	(+262%)	92.7	(+18%)	322.4	(+273%)	62.1	(+41%)
	C	278.8	(+97%)	210.5	(+169%)	256.2	(+196%)	201.2	(+356%)
	QS	655.1	(+362%)	171.0	(+118%)	408.8	(+373%)	106.3	(+141%)
	QC	420.5	(+197%)	288.8	(+269%)	342.6	(+296%)	245.3	(+456%)
	SC	792.2	(+459%)	303.2	(+287%)	578.6	(+570%)	263.2	(+496%)
	QSC	933.9	(+559%)	381.5	(+387%)	665.0	(+670%)	307.4	(+596%)
compressed	Q	56.0		32.0		33.4		16.8	
	S	139.3	(+149%)	34.1	(+7%)	95.3	(+185%)	22.8	(+35%)
	C	56.6	(+1%)	44.5	(+39%)	52.7	(+58%)	42.7	(+154%)
	QS	195.3	(+249%)	66.1	(+107%)	128.7	(+285%)	39.7	(+135%)
	QC	112.6	(+101%)	76.5	(+139%)	86.1	(+158%)	59.6	(+254%)
	SC	195.9	(+250%)	78.6	(+146%)	148.0	(+343%)	65.5	(+289%)
	QSC	251.9	(+350%)	110.6	(+246%)	181.4	(+443%)	82.4	(+389%)
# suggestions		6,382,973		3,484,172		4,075,725		2,118,571	

comprising a sample $\mathbf{x} = \{\mathbf{x}_j\}_{j=1}^{m}$ of m suggestions mined for q, as discussed in the previous section. Each element $\mathbf{x}_j = \Phi(q_j, q)$ in the sample is a vector representation of a candidate suggestion q_j, according to the feature extractor Φ. In Section 3.3, we will describe the various features used in our investigation, including query-dependent and query-independent ones.

The output space $\mathcal{Y}$ for our learning problem contains a set of ground-truth labels $\mathbf{y} = \{y_j\}_{j=1}^{m}$. In order to target the learning process towards identifying effective query suggestions, each label y_j is automatically defined based on the observed retrieval effectiveness e_j of the search results produced for the query suggestion q_j, according to:

$$y_j = \begin{cases} 3 : \text{if } e_j > e, \\ 2 : \text{if } e_j = e, \\ 1 : \text{if } 0 < e_j < e, \\ 0 : \text{otherwise}, \end{cases} \tag{1}$$

where $e = \Delta_r(\mathcal{R}(q)|q, n_r)$ and $e_j = \Delta_r(\mathcal{R}(q_j)|q, n_r)$ denote the retrieval performance at rank n_r (given by any standard information retrieval evaluation metric Δ_r, such as nDCG [27]) attained by the ranking $\mathcal{R}(\bullet)$ produced by a reference retrieval system for a given input (i.e., the query q or its query suggestion q_j).

Lastly, we must define a loss function to guide our learning process. In this article, we adopt two listwise learning to rank approaches [8,31], which directly

Table 2 All features computed in this work for each query suggestion q_j.

	Feature	Description	Total
query-dependent	BM25	Full and per-field BM25 score [37]	4
	LM	Full and per-field LM score [49]	4
	DPH	Full and per-field DPH score [3]	4
	PL2	Full and per-field PL2 score [2]	4
	MQT	Full and per-field MQT score	4
	pBiL	Full pBiL score [34]	1
	MRF	Full MRF score [32]	1
query-independent	Tokens	Full and per-field token count	4
	Terms	Fraction of unique terms in q_j	1
	Chars	Number of characters in q_j	1
	RepChars	Length of longest repeated substring in q_j	2
	Digits	Number of digits in q_j	2
	Punctuation	Number of punctuation characters in q_j	2
	Badwords	Presence, number, and fraction of swearing in q_j	3
	UrlFragments	Whether q_j contains or is a URL	2
	Clicks	Total number of clicked results for q_j	1
	Sessions	Total number of sessions with q_j	1
	SessionClicks	Mean, max, and s.d. clicks on q_j per session	3
	SessionLength	Mean, max, and s.d. length of sessions with q_j	3
	SessionPosition	Mean, max, and s.d. relative position of q_j per session	3
	SessionSuccess	Ratio of clicked queries in sessions with q_j	1
	Grand total		51

leverage an information retrieval evaluation metric as their loss function. In particular, we define $\Delta_s(\mathcal{Q}(q)|q, n_s)$ as the loss at rank n_s of retrieving the suggestions $\mathcal{Q}(q)$ in response to the query q. Note that, different from the *search results* evaluation metric Δ_r used to define our ground-truth labels in Equation (1), this metric is used to evaluate rankings of *query suggestions*. Our experimental setup choices for the sample size m, labelling function Δ_r and cutoff n_r, loss function Δ_s and cutoff n_s, and learning algorithms are fully described in Section 5.3.

3.3 Query Suggestion Features

Having discussed alternative approaches for sampling candidate suggestions from a query log and how to learn an effective ranking function for a given sample, we now describe the features used to represent each suggestion in the learning process. As summarised in Table 2, we broadly organise all query suggestion features used by our approach as either query-dependent or query-independent, according to whether they are computed on-the-fly at querying time or offline at indexing time, respectively. While the considered query-dependent features are standard features commonly used in the literature for learning to rank for web search [29], the query-independent ones are specifically proposed here to estimate the quality of different candidate suggestions.

Given a query q, query-dependent features are directly computed by scoring the occurrences of the terms of q in each of the fields of each candidate suggestion. To do so, we use multiple weighting schemes implemented in Terrier [33],[3] including standard weighting models, such as BM25 [37], language modelling with Dirichlet smoothing (LM) [49], the Divergence From Randomness (DFR) DPH [3] and PL2 [2] models, and a count of matching query terms (MQT). Additionally, we employ term dependence models based on Markov Random Fields (MRF [32]) and the DFR framework (pBiL [34]), which highly score suggestions where the query terms co-occur in close proximity. All features can be efficiently computed at querying time with a single pass over the posting lists for the query q in the index of structured virtual documents.

As for query-independent features, they are all computed at indexing time. In particular, we consider features that can be directly estimated from the query log itself, so as to draw insights regarding which query log evidence is helpful for ranking query suggestions. These include quality features, such as the length of the suggestion in tokens and characters (too long suggestions may denote robot-submitted queries) and the presence of digits, punctuation, and swearing (which usually indicate low-quality or adult-oriented queries). Additionally, we also derive features measuring the popularity of a suggestion in terms of number of sessions and clicks, as popular suggestions arguably indicate higher quality a priori. Finally, we consider features that summarise the profile of a suggestion across the sessions where it occurs. These include the number of clicks received, the total number of queries and the ratio of clicked queries, and the suggestion's relative position in each session.

4 Evaluating Query Suggestions

The effectiveness of a query suggestion mechanism is typically assessed in a subjective manner, based on user studies [42]. On the other hand, Hauff et al. [25] have shown that users are not good at predicting the retrieval performance of query suggestions. At the same time, it seems natural to assess the performance of a query suggestion in terms of how much it helps users to satisfy their information need. More precisely, we argue that the effectiveness of a query suggestion mechanism should be assessed as to whether its suggested queries help the users satisfy the information need expressed by their initial query. With this in mind, we formalise a framework for the quantitative evaluation of query suggestions that directly builds upon existing efforts for information retrieval evaluation. In particular, we envisage two scenarios, depending on whether or not the users' initial query is ambiguous.

The first scenario assumes that the user's query is unambiguously defined. In this scenario, given a query q and a ranking of suggestions $\mathcal{Q}(q)$ produced for this query, our goal is to evaluate these suggestions in terms of their retrieval performance when used as a replacement for q. In particular, we introduce

[3] `http://terrier.org`

s-eval$_\Psi(\bullet)$ for query suggestion evaluation as the counterpart of a standard retrieval evaluation metric *eval*$(\bullet)$ (e.g., nDCG [27]), according to:

$$s\text{-}eval_\Psi(\mathcal{Q}(q)|q,k,n) = \Psi_{j=1}^{k}\big[eval(\mathcal{R}(q_j)|q,n)\big], \tag{2}$$

where k is the number of top suggestions to be evaluated (the *suggestion* evaluation cutoff), n is the number of top results to be evaluated for each suggestion (the *retrieval* evaluation cutoff), $\mathcal{R}(q_j)$ is the ranking produced for the query suggestion q_j by a reference retrieval system, and Ψ is a summary statistic. In Section 6, we report both the maximum (Ψ = 'max') and the average (Ψ = 'avg') retrieval performance attained by the top k suggested queries. For instance, using s-nDCG$_\Psi$@k,10 with Ψ = 'max' and $k = 1$, we can evaluate the effectiveness (in terms of the nDCG@10 performance of the resulting ranking of search results) of a query suggestion mechanism at providing a single suggestion. Such a suggestion could be used, e.g., for automatically reformulating the initial query. With Ψ = 'avg' and $k = 8$, we can model a typical application of query suggestions, as seen on the search box of modern web search engines. Note that both the Ψ = 'max' and Ψ = 'avg' summary statistics consider the top k suggested queries as an unordered set, regardless of how these suggestions were ranked with respect to each other. Although rank-based summary statistics are certainly possible, this would imply assuming that users prefer the top ranked suggestion over the others. Since, to the best of our knowledge, there is no empirical study supporting this assumption, we opted for a set-based evaluation in our investigations.

The query suggestion evaluation metrics generated by Equation (2) assume that the query q unambiguously expresses the user's information need. Indeed, both q and the suggestion q_j are evaluated with respect to the information need represented by q. In practice, however, the queries submitted to a web search engine are often ambiguous [43], with the same query being used by different search users to represent different information needs [45]. In this situation, providing a diverse list of suggestions could not only help the users better specify their need, but would also enable an effective diversification of the search results, as mentioned in Section 2. To cater for query ambiguity, we formulate a second scenario within our evaluation framework to quantitatively assess the diversity of the suggestions produced by a given mechanism. Analogously to the definition in Equation (2), we introduce *s-deval*$(\bullet)$ for query suggestion evaluation as the counterpart of a diversity evaluation metric *deval*$(\bullet)$ (e.g., α-nDCG [17]), according to:

$$s\text{-}deval(\mathcal{Q}(q)|q,k,n) = deval(\mathcal{D}(q,\mathcal{Q}(q),k)|q,n), \tag{3}$$

where $\mathcal{D}(q,\mathcal{Q}(q),k)|q,n)$ is the ranking of search results produced by a reference diversification system for the query q using the top k produced suggestions from $\mathcal{Q}(q)$, and n is the depth at which this ranking should be evaluated. For instance, s-α-nDCG@8,10 measures the search result diversification performance (in terms of α-nDCG@n, with $n = 10$) of the top $k = 8$ suggestions produced by a given mechanism, when used as input to a diversification approach, such as xQuAD [38], as mentioned in Section 2.

With the proposed framework, using a fixed reference retrieval system, we can quantitatively compare the suggestions produced by different mechanisms with respect to one another, as well as with respect to the retrieval effectiveness attained by the initial query alone (i.e., $eval(\mathcal{R}(q)|q,n)$). Likewise, we can also contrast the diversification performance of different suggestion mechanisms in contrast to one another, as well as in comparison to the diversification performance of the initial query (i.e., $deval(\mathcal{R}(q)|q,n)$). In the next sections, we leverage this framework to assess the effectiveness of our proposed learning to rank approach as well as of state-of-the-art query suggestion baselines at providing query suggestions for both adhoc and diversity search.

5 Experimental Setup

This section details the experimental setup that supports the investigations in this article. In particular, we aim to answer four main research questions:

1. *How effective is our query suggestion approach for adhoc search?*
2. *How effective is our query suggestion approach for diversity search?*
3. *Which features (from Table 2) are useful for ranking query suggestions?*
4. *How robust to missing relevance assessments is our evaluation framework?*

In the remainder of this section, we describe the test collection and retrieval baselines used in our investigation, as well as the training procedure carried out to enable our proposed learning to rank approach for query suggestion.

5.1 Test Collection

Our experiments are based on the evaluation paradigm provided by the TREC 2009, 2010, and 2011 Web tracks [13,15,16]. The TREC Web track provides a test collection comprising a total of 148 web search queries and corresponding relevance judgements to enable the assessment of adhoc and diversity search approaches. As a document corpus, this track uses the ClueWeb09 corpus,[4] comprising over one billion web documents. Following the standard procedure at TREC, we use the category A portion of ClueWeb09, comprising over 500 million English documents. To retrieve suggestions for each of the 148 TREC Web track queries, we use the MSN 2006 query log, a one-month log with 15 million queries submitted by US users to MSN Search (now Bing) during spring 2006.[5] We index the structured virtual documents produced from the MSN 2006 query log using Terrier [33] with positional information, so as to enable the extraction of proximity features. In particular, we apply Porter's weak stemming and do not remove stopwords. Finally, sessions are determined using a standard 30 min timeout. In addition, sessions with more than 50 queries

[4] `http://boston.lti.cs.cmu.edu/Data/clueweb09/`

[5] `http://research.microsoft.com/en-us/um/people/nickcr/wscd09`

Table 3 Salient statistics of the test collection used in our experiments.

query log	MSN 2006	corpus	ClueWeb09
#queries	14,921,285	#queries	148
#unique queries	6,623,635	#documents	503,903,810
#sessions	7,470,915	#relevants (adhoc)	15,248
#clicks	12,251,067	#relevants (diversity)	16,525

are discarded, as they are likely produced by robots [42]. Salient statistics of the MSN 2006 query log and the ClueWeb09 corpus are provided in Table 3.

Candidate suggestions are evaluated with respect to their performance at ranking documents from the ClueWeb09 corpus. To this end, we use the Bing Search API as the reference adhoc retrieval system, by directly evaluating its returned URLs against those judged relevant in this corpus.[6] While using the Bing API provides a state-of-the-art reference retrieval system and is efficient enough to enable the large-scale evaluation conducted in this article, the rankings produced by using this API should be seen as a crude approximation of what Bing could achieve if restricted to searching only the ClueWeb09 corpus in the first place [39]. Nonetheless, Clarke et al. [13,15] have shown that rankings produced by a commercial search engine outperform almost all submitted runs in the TREC 2009 and 2010 Web tracks. Finally, as the reference diversification system, we employ the state-of-the-art xQuAD diversification framework [38]. As described in Section 2, xQuAD was the top-performing among the officially submitted approaches in the diversity task of the TREC 2009, 2010, and 2011 Web tracks [13,15,16], and is hence a representative of state-of-the-art diversification approaches.

5.2 Query Suggestion Baselines

To answer our first two research questions, we compare our proposed approach to two baselines. The first of these is the approach of Broccolo et al. [7], which served as the basis for the suggestion representation adopted in this work, as described in Section 3.1. As discussed in Section 2, their approach was shown to perform at least as effectively as the state-of-the-art query-flow graph approach of Boldi et al. [6] for head queries, while consistently outperforming it for queries with little or no past evidence in the MSN 2006 query log. Hence, it is used here as a representative of state-of-the-art query suggestion mechanisms. Additionally, we compare both our approach and that by Broccolo et al. to the query suggestions produced by the Bing Suggestion API.[7] While Bing can suggest queries not present in our test query logs (and arguably has suggestion models built from much larger query logs), this provides a reference performance for an industrial-strength suggestion mechanism.

[6] All rankings were obtained in February 2012 using Bing API v2.0.

[7] All query suggestions were obtained in February 2012 using Bing API v2.0.

5.3 Training Procedure

To enable our learning approach, we first produce a set of ground-truth suggestions for each test query. Following the scheme formalised in Equation (1), we automatically label a pool of 105,325 suggested queries, comprising the union of suggestions retrieved using the different sampling strategies discussed in Section 3.1 (e.g., suggestions retrieved based on different fields, or those that come from satisfactory sessions). For the labelling function Δ_r in Equation (1), we use nDCG@10, which is a typical target in a web search setting [26]. For our learning setup, following common practice [35], we consider a sample of $m = 1000$ suggestions retrieved for a given query using BM25 [37]. As a loss function, we use nDCG@100. Such a deeper cutoff provides a more informative guidance to the learning process, by capturing swaps between relevant and non-relevant documents beyond our target evaluation cutoff ($n = 10$) [36]. As learning algorithms, we employ two listwise learning to rank approaches: Automatic Feature Selection (AFS [31])—a greedy algorithm which has been shown to perform effectively for learning to rank for web adhoc and diversity search [40]—and LambdaMART [8, 24]—a boosted regression tree algorithm, which ranked first at Track 1 of the recent Yahoo! Learning To Rank Challenge [11]. In order to learn effective query suggestion rankings, we perform a 5-fold cross validation, by splitting the available queries into training (60%), validation (20%), and test (20%) sets. Accordingly, our results are reported on the test queries across all folds.

6 Experimental Results

In this section, we discuss our experimental results regarding the three research questions introduced in Section 5.

6.1 Adhoc Retrieval Performance

To evaluate the effectiveness of our query suggestion approach for adhoc search, we analyse both the impact of the initial sample of candidate suggestions (Section 6.1.1), as well as the improvements brought by our learning to rank approach (Section 6.1.2). Lastly, we analyse the effectiveness of our approach for queries with various frequencies in the query logs (Section 6.1.3), so as to assess the impact of data sparsity.

6.1.1 Sampled Suggestions

Before evaluating our learning approach to query suggestion, we assess the alternative choices introduced in Section 3.1 for producing suggestion samples. In particular, we analyse this question in light of three orthogonal dimensions. The first dimension concerns the sessions from which to mine candidate suggestions: all sessions vs. satisfactory sessions (i.e., those with a click in the last

query). The second dimension concerns the queries from a given session to be indexed as candidate suggestions: all queries vs. the last one. Finally, the third dimension relates to the sources of evidence to index as fields for each candidate suggestion: the suggestion itself (Q), its co-session queries (S), its queries with a common click (C), or any combination of these three fields.

In order to assess the full potential of each of these sampling alternatives, Table 4 summarises their performance in terms of the number of relevant suggestions retrieved at maximum recall depth (i.e., RelRet@1000). For this investigation, relevance labels are defined as in Equation (1).[8] A ▼ symbol denotes a significantly worse performance compared to the best result in the same column (highlighted in bold), according to a paired t-test with $p < 0.01$. The lack of a significant difference is denoted by the ∘ symbol. From the table, regarding our first dimension of interest, we observe that indexing only queries from satisfactory sessions (as opposed to all sessions) almost always leads to improved performance. This corroborates the findings reported by Broccolo et al. [7], by showing that such sessions are more likely to contain effective suggestions. Nonetheless, regarding our second considered dimension, we observe that indexing only the last query in a session, as proposed by Broccolo et al., substantially decreases performance, regardless of whether this session is satisfactory or not. Lastly, regarding our third dimension of interest, we observe a natural increase in recall as we combine more fields together, with QSC being the overall best combination. This shows that click evidence further improves the QS combination used by Broccolo et al. [7]. On the other hand, taking into account the performance of individual fields can also be beneficial. Indeed, as shown in the table, the Q field is the most effective, with S and C showing a similar performance. Recalling our first research question, on the effectiveness of our proposed query suggestion approach for adhoc search, we conclude that mining query suggestions among all queries in satisfactory sessions, and considering a multi-field representation (i.e., the QSC combination), particularly with the added click evidence, provides the most effective sampling to be used for learning to rank query suggestions.

6.1.2 Learned Suggestions

After investigating alternative strategies for building an initial sample of query suggestions in response to a query, we analyse whether this sample can be further improved by our learning to rank approach. In particular, we focus on the most promising samples identified by our previous experiments, namely, those comprising all queries from satisfactory sessions (i.e., BM25(QSC) in Table 4). For this investigation, we instantiate our proposed evaluation framework described in Section 4 and report our results in terms of s-nDCG$_\Psi$@k,10—i.e., the summary ('max' or 'avg') retrieval performance (in terms of the standard

[8] Note that suggestions with a relevance label 1 (i.e., with a positive yet lower retrieval effectiveness than that attained by the initial query) are also considered, as they may bring useful evidence for the diversification scenario addressed in Section 6.2.

Table 4 Query suggestion ranking performance (in terms of RelRet@1000) attained by alternative sampling strategies. These encompass both different suggestion representations based on the query (Q), co-session (S), and co-clicked (C) fields, as well as different choices of whether to consider suggestions from all sessions or only the satisfactory ones, and whether to consider all or only the last suggestion in each session. Relevance labels are defined as in Equation (1). The representation used by Broccolo et al. [7] is marked with a † symbol.

sessions	all		satisfactory	
queries	all	last	all	last
BM25(Q)	73▼	68▼	76▼	69▼
BM25(S)	68▼	55▼	72▼	58▼
BM25(C)	60▼	59▼	61▼	60▼
BM25(QS)	96▼	91▼	102▼	†94°
BM25(QC)	92▼	85▼	91▼	83▼
BM25(SC)	104▼	84▼	105▼	82▼
BM25(QSC)	**115**	**101**	**117**	**98**

nDCG@10) attained by the top k ranked suggestions. As discussed in Section 4, we test two values of k, representing two distinct scenarios. In the first scenario, we set $k = 1$, which assesses the effectiveness of each query suggestion mechanism at providing a single suggestion that could be used, e.g., for an automatic reformulation for the initial query. In the second scenario, we set $k = 8$, which is the maximum number of suggestions retrieved by the Bing Suggestions API as well as by the search interfaces of current commercial web search engines, and hence represents a typical application of query suggestion.

Table 5 shows the results of this investigation. In each cell, significant improvements with respect to the initial query, Bing Suggestions, and the BM25 unsupervised baseline of Broccolo et al. [7] are denoted by a superscript q, b, and u, respectively. As before, significance is verified using a paired t-test. The symbols $\triangle$ ($\triangledown$) and ▲ (▼) denote a significant increase (decrease) at the $p < 0.05$ and $p < 0.01$ levels, respectively, while $\circ$ denotes no significant difference. Firstly, compared to the approach of Broccolo et al. [7] (i.e., the BM25 entries in Table 5), our learning to rank approach using either AFS or LambdaMART consistently improves, with significant gains in many settings. Indeed, when retrieving multiple suggestions (i.e., $k = 8$), the suggestions produced by our approach are comparable to those provided by the Bing API. In addition, significant gains are observed for the task of returning a single highly effective suggestion for query reformulation (i.e., $k = 1$). As discussed in Section 5.2, this is a remarkable result, particularly since the Bing API is allowed to return effective suggestions not present in our one-month-long query log, while arguably making use of much larger logs. Lastly, compared to the initial query, no query suggestion mechanism improves for $k = 1$, including Bing's own top ranked suggestion. This shows that an automatic reformulation of the initial query using the top suggestion would be risky. However, for $k = 8$ (Ψ = 'max'), both Bing and our learning approach are able to suggest at least one query that outperforms the initial one in some settings, although not significantly. Overall, the results in this section answer our first research

Table 5 Adhoc retrieval performance (in terms of s-nDCG$_\Psi$@k,10) attained by the top k suggestions produced by various query suggestion mechanisms. As described in Equation (2), Ψ = 'max' and Ψ = 'avg' denote, respectively, the maximum and the average performance attained by the search results produced by each of the k suggestions in terms of nDCG@10. Our learning to rank approach is deployed using either AFS [31] or LambdaMART [8] to rerank the initial sample produced by BM25 using different field combinations. The latter corresponds to the unsupervised (u) approach of Broccolo et al. [7], which is used as a baseline in the table. The retrieval performances of the original query (q) and that of Bing Suggestions (b) are provided as additional reference values.

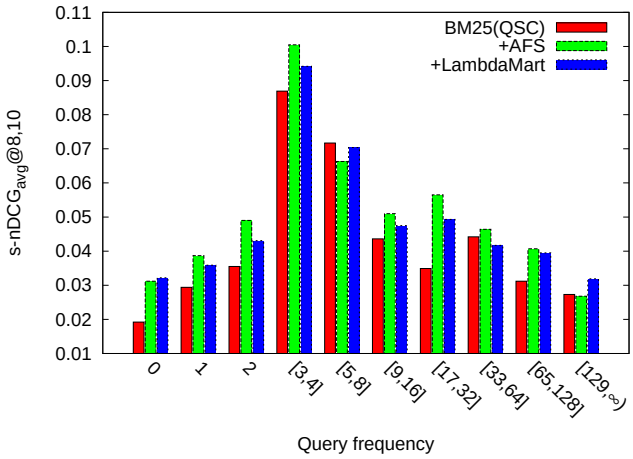
	k=1	k=8	
	Ψ = 'max'	Ψ = 'max'	Ψ = 'avg'
Query only	**0.1151**	0.1151	**0.1151**
Bing Suggestions	0.0485$^{(q_\circ)}$	**0.1188**$^{(q_\circ)}$	0.0447$^{(q_\circ)}$
BM25(Q)	0.0644$^{(q_\circ b_\circ)}$	0.1062$^{(q_\circ b_\circ)}$	0.0392$^{(q_\circ b_\circ)}$
+AFS	0.0224$^{(q_\circ b_\circ u_\circ)}$	0.0430$^{(q_\circ b_\circ u_\circ)}$	0.0136$^{(q_\circ b_\circ u_\circ)}$
+LambdaMART	0.0736$^{(q_\circ b_\blacktriangle u_\circ)}$	0.1123$^{(q_\circ b_\circ u_\circ)}$	0.0452$^{(q_\circ b_\circ u_\triangle)}$
BM25(S)	0.0205$^{(q_\circ b_\circ)}$	0.0715$^{(q_\circ b_\circ)}$	0.0179$^{(q_\circ b_\circ)}$
+AFS	0.0675$^{(q_\circ b_\triangle u_\circ)}$	0.1033$^{(q_\circ b_\circ u_\circ)}$	0.0381$^{(q_\circ b_\circ u_\circ)}$
+LambdaMART	0.0574$^{(q_\circ b_\circ u_\circ)}$	0.0962$^{(q_\triangledown b_\triangledown u_\circ)}$	0.0349$^{(q_\circ b_\triangledown u_\circ)}$
BM25(C)	0.0452$^{(q_\circ b_\circ)}$	0.0808$^{(q_\circ b_\blacktriangledown)}$	0.0319$^{(q_\circ b_\blacktriangledown)}$
+AFS	0.0722$^{(q_\circ b_\blacktriangle u_\circ)}$	0.0971$^{(q_\circ b_\circ u_\circ)}$	0.0387$^{(q_\circ b_\circ u_\circ)}$
+LambdaMART	0.0632$^{(q_\circ b_\circ u_\circ)}$	0.0948$^{(q_\triangledown b_\triangledown u_\circ)}$	0.0389$^{(q_\circ b_\circ u_\circ)}$
BM25(QS)	0.0429$^{(q_\circ b_\circ)}$	0.0912$^{(q_\blacktriangledown b_\blacktriangledown)}$	0.0296$^{(q_\circ b_\circ)}$
+AFS	0.0904$^{(q_\circ b_\circ u_\circ)}$	0.1145$^{(q_\circ b_\circ u_\circ)}$	0.0475$^{(q_\circ b_\circ u_\circ)}$
+LambdaMART	0.0745$^{(q_\circ b_\blacktriangle u_\circ)}$	0.1126$^{(q_\circ b_\circ u_\circ)}$	0.0456$^{(q_\circ b_\circ u_\circ)}$
BM25(QC)	0.0475$^{(q_\circ b_\circ)}$	0.0909$^{(q_\triangledown b_\triangledown)}$	0.0377$^{(q_\circ b_\circ)}$
+AFS	0.0863$^{(q_\circ b_\circ u_\circ)}$	0.1125$^{(q_\circ b_\circ u_\blacktriangle)}$	0.0448$^{(q_\circ b_\circ u_\blacktriangle)}$
+LambdaMART	0.0771$^{(q_\circ b_\blacktriangle u_\circ)}$	0.1091$^{(q_\circ b_\circ u_\triangle)}$	0.0458$^{(q_\circ b_\circ u_\blacktriangle)}$
BM25(SC)	0.0456$^{(q_\circ b_\circ)}$	0.0839$^{(q_\blacktriangledown b_\blacktriangledown)}$	0.0341$^{(q_\circ b_\triangledown)}$
+AFS	0.0817$^{(q_\circ b_\blacktriangle u_\circ)}$	0.1077$^{(q_\circ b_\circ u_\blacktriangle)}$	0.0435$^{(q_\circ b_\circ u_\circ)}$
+LambdaMART	0.0659$^{(q_\circ b_\triangle u_\circ)}$	0.1054$^{(q_\circ b_\circ u_\blacktriangle)}$	0.0449$^{(q_\circ b_\circ u_\circ)}$
BM25(QSC)	0.0501$^{(q_\circ b_\circ)}$	0.0985$^{(q_\triangledown b_\circ)}$	0.0384$^{(q_\circ b_\circ)}$
+AFS	0.0852$^{(q_\circ b_\circ u_\circ)}$	0.1166$^{(q_\circ b_\circ u_\circ)}$	0.0474$^{(q_\circ b_\circ u_\circ)}$
+LambdaMART	0.0778$^{(q_\circ b_\circ u_\circ)}$	0.1100$^{(q_\circ b_\circ u_\blacktriangle)}$	0.0459$^{(q_\circ b_\circ u_\circ)}$

question, by attesting the effectiveness of our approach compared to the state-of-the-art query suggestion approach of Broccolo et al. [7] as well as to the industrial-strength suggestion mechanism provided by the Bing API.

6.1.3 Performance under Sparsity

To complete the investigations of our first research question, we analyse the impact of query sparsity on the effectiveness of our proposed learning to rank approach to query suggestion. In particular, an inherited characteristic of the query suggestion representation adopted by our approach is its resilience to sparse data. As discussed in Section 3.1, most existing query suggestion approaches suffer when there is limited session or click information for a given

Fig. 2 Suggestion adhoc effectiveness (in terms of s-nDCG$_{avg}$@8,10) for queries with different frequency in the MSN 2006 query log. Frequencies are split into exponentially-sized bins, so that the number of queries in each bin is roughly balanced. The learning variants of our approach using either AFS [31] or LambdaMART [8] are used to rerank the initial sample produced by the unsupervised approach of Broccolo et al. [7] using BM25 on all fields (i.e., QSC), which serves as a baseline in the figure.



query. Instead, by indexing candidate suggestions at the term level, our approach improves the chance of identifying at least one of these suggestions as a potentially relevant match for even an unseen query, provided that the query and the suggestion share at least one term. To illustrate this behaviour, Figure 2 breaks down the performance of our query suggestion approach for queries with different frequencies in the MSN 2006 query log. As shown in the figure, both of our learning to rank variants as well as the approach of Broccolo et al. [7] are able to provide effective suggestions even for completely unseen queries (i.e., queries with a zero frequency in the query log). While this resilience to sparsity comes mostly from the structured virtual document representation inspired by the approach of Broccolo et al., it is interesting to note that our learning variants further improve on top of their approach for almost the entire range of query frequencies. This further attests the effectiveness of our proposed approach, corroborating the findings in Section 6.1.2.

6.2 Diversification Performance

Besides assessing the effectiveness of our produced suggestions in terms of their attained adhoc retrieval performance, in this section, we address our second research question, by assessing the effectiveness of the produced suggestions when used for diversifying the search results. As discussed in Section 4, for this evaluation, we use three different instantiations of the *s-deval* metric defined in Equation (3), by leveraging the three primary metrics for diversity search

evaluation used in the TREC Web track [15]: α-nDCG [17], ERR-IA [12], and NRBP [18]. These metrics implement a cascade user model [14], which assumes an increasing probability that the users will stop inspecting the search ranking once they find a relevant result. Consequently, they reward diversity and penalise redundancy in the ranking. Analogously to the previous evaluations, we consider the scenario where a user would inspect the top $n = 10$ results, diversified by the xQuAD framework (as the reference diversification system) using the top k suggestions provided by each query suggestion mechanism. Table 6 shows the results of this investigation, with the aforementioned symbols denoting significant differences (or lack thereof).

Table 6 Diversification performance (for various s-$deval@k, n$ metrics; see Equation (3)) attained by the top $n = 10$ search results ranked by the xQuAD diversification framework [38] using the top $k = 8$ suggestions produced by various query suggestion mechanisms. Suggestions produced by the unsupervised (u) approach of Broccolo et al. [7] (i.e., BM25(QSC)) serve as a baseline in the table. The diversification performances of the original query (q) and that of Bing Suggestions (b) are provided as additional reference values.

	s-α-nDCG@8,10	s-ERR-IA@8,10	s-NRBP@8,10
Query only	0.4832	0.3759	0.3452
Bing Suggestions	$0.4960^{(q_\circ)}$	$0.3917^{(q_\circ)}$	$0.3657^{(q_\triangle)}$
BM25(QSC)	$0.4862^{(q_\circ b_\circ)}$	$0.3792^{(q_\circ b_\circ)}$	$0.3515^{(q_\circ b_\circ)}$
+AFS	$0.4893^{(q_\circ b_\circ u_\circ)}$	$0.3833^{(q_\circ b_\circ u_\circ)}$	$0.3560^{(q_\circ b_\circ u_\circ)}$
+LambdaMART	$\underline{\mathbf{0.4970}}^{(q_\triangle b_\circ u_\triangle)}$	$\underline{\mathbf{0.3919}}^{(q_\triangle b_\circ u_\triangle)}$	$\underline{\mathbf{0.3659}}^{(q_\triangle b_\circ u_\triangle)}$

From Table 6, we once again observe that our approach consistently outperforms the state-of-the-art baseline suggestion mechanism of Broccolo et al. [7], with significant improvements when using LambdaMART. Indeed, only Bing (for s-NRBP) and LambdaMART (for all metrics) can significantly outperform the initial query. Moreover, the attained performance of our approach does not differ significantly from the performance attained by the suggestions produced by the Bing API. Once again, this is a remarkable result, given the substantially larger amount of data available to Bing compared to our one-month query log snapshot. Overall, this answers our second research question, by showing that our learning approach is also effective at providing query suggestions to be used for search result diversification.

6.3 Feature Analysis

In order to address our third research question, we investigate which features are effective for learning to rank query suggestions. In particular, Table 7 lists the top 10 query-dependent and top 10 query-independent features, selected according to their correlation (Pearson's r) with the training labels, as defined in Equation (1). From the table, we observe that the overall top 10 features are all query-dependent, showing both the topical nature of this task, and the

benefit of leveraging evidence from multiple sources in a query log (i.e., the Q, S, and C fields), with an aggregation of all available evidence (i.e., the QSC combination) performing the best. As for the top query-independent features, our learned query suggestion models generally benefit from lexical features, such as the number of punctuation symbols in a candidate suggestion, or its total length in characters or tokens. In addition, features based on past usage behaviour, including session and click information, are also effective.

Table 7 Top 10 query-dependent and query-independent features for learning to rank suggestions. Features are ranked by their correlation (Pearson's r) with the learning labels.

Query-dependent features			Query-independent features		
Rank	Feature	r	Rank	Feature	r
1	BM25(QSC)	0.1699	11	Punctuation (fraction)	0.0593
2	PL2(QSC)	0.1595	14	Punctuation (total)	0.0505
3	MQT(QSC)	0.1117	16	SessionLength (mean)	0.0490
4	LM(QSC)	0.0908	21	SessionLength (max)	0.0415
5	BM25(Q)	0.0864	25	Chars	0.0230
6	DPH(QSC)	0.0826	26	SessionClicks (mean)	0.0229
7	MQT(Q)	0.0799	28	SessionClicks (max)	0.0195
8	PL2(Q)	0.0757	29	Tokens(S)	0.0191
9	PL2(S)	0.0715	31	Clicks	0.0173
10	pBiL(QSC)	0.0632	32	SessionClicks (s.d.)	0.0137

6.4 Robustness to Missing Relevance Assessments

As discussed in Section 4, our evaluation framework leverages document relevance assessments from the adhoc and diversity test collections of the TREC Web track [13,15,16]. As a result, the robustness of our proposed framework directly depends on its ability to reuse the relevance assessments from these test collections. In particular, in our framework, the diversity document relevance assessments produced for a given query are directly reused to evaluate a different document ranking produced for the same query, namely, the ranking produced by using query suggestions as input to a reference diversification approach, such as xQuAD [38]. On the other hand, the adhoc relevance assessments for a query are reused to assess the effectiveness of a ranking produced for different queries, i.e., each of the suggestions produced for the initial query. The latter scenario explicitly assumes a user with a clearly specified information need, hence considering query suggestion as the task of identifying effective replacements for the original query. Nonetheless, the effectiveness of such replacement queries may be underestimated in our evaluation framework, simply because they can retrieve documents that were not judged at all for the initial query. To address our fourth and last research question, Table 8 shows the extent to which missing relevance assessments impact the reusability of the TREC 2009, 2010, and 2011 Web track assessments within

Table 8 Ratio of judged (J@10) and relevant (P@10) documents among the top 10 documents retrieved by Bing for each of the suggestions included in the BM25(QSC) sample for each query. Per-query figures are summarised by multiple statistics and broken down according to the considered adhoc ($k = 1$ and $k = 8$) and diversity ($k = 8$) search scenarios. As a baseline for measuring the reusability of the TREC Web track relevance assessments, J@10 and P@10 figures attained by document rankings produced for the initial query by Bing and by a standard BM25 formulation are shown in the bottom half of the table.

suggestions	adhoc						diversity	
BM25(QSC)	k=1		k=8				k=8	
	Ψ = 'max'		Ψ = 'max'		Ψ = 'avg'			
	J@10	P@10	J@10	P@10	J@10	P@10	J@10	P@10
average	0.0344	0.0208	0.5060	0.3120	0.2268	0.1322	0.5647	0.3927
median	0.0000	0.0000	0.5000	0.3000	0.2000	0.0750	0.6000	0.4000
std. dev.	0.1373	0.0962	0.2774	0.2641	0.1786	0.1439	0.2338	0.2350
minimum	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
maximum	0.9000	0.9000	0.9000	0.9000	0.7750	0.6875	0.9000	0.9000

query only	adhoc		diversity	
	J@10	P@10	J@10	P@10
Bing	0.5893	0.3400	0.5627	0.3980
BM25	0.3291	0.0973	0.2493	0.0899

our framework. In particular, we consider both the number of judged (J@10) and relevant (P@10) documents among the top 10 documents retrieved for each suggestion in the BM25(QSC) sample, which served as the basis for most query suggestion mechanisms investigated in Sections 6.1 through 6.3. These figures are compared to a BM25 ranking for the initial query, which represents a typical use case of reuse of the TREC Web track assessments for evaluation. For comparison, we also include a ranking produced by Bing, which was used as the reference retrieval system in this article, as discussed in Section 5.1.

Firstly, from the bottom half of Table 8 (the 'query only' half), we observe that, despite not being restricted to retrieve documents from the ClueWeb09 corpus, Bing attains a much higher coverage of judged (J@10) and relevant (P@10) documents than BM25 (adhoc: Bing's J@10 = 0.5893, P@10 = 0.3400 vs. BM25's J@10 = 0.3291, P@10 = 0.5893; diversity: Bing's J@10 = 0.5627, P@10 = 0.3980 vs. BM25's J@10 = 0.2493, P@10 = 0.0899). This observation highlights the importance of having a high performing reference retrieval system for evaluating the effectiveness of query suggestions in large web corpora such as ClueWeb09. Moreover, it corroborates our choice in Section 5.1 for using the API of a commercial web search engine for this purpose.

Next, with specific regards to our fourth research question, on the robustness of our evaluation framework to missing assessments, from the top half of Table 8 (the 'suggestions' half), we observe that most of the considered search scenarios show a reasonable coverage of the relevance assessments leveraged from the TREC 2009, 2010, and 2011 Web tracks. In particular, assessing the effectiveness of a set of suggestions (adhoc scenario, $k = 8$) shows

a high robustness to missing assessments, with a coverage of judged (J@10) and relevant (P@10) documents that compares favourably to that attained by a standard BM25 ranking produced for the initial query (average J@10 up to 0.5060 vs. BM25's 0.3291; average P@10 up to 0.3120 vs. BM25's 0.0973). The diversity scenario, in turn, shows an even higher reuse of the underlying document relevance assessments, with average figures of J@10 = 0.5647 and P@10 = 0.3927. Such a higher coverage is due to the fact that this scenario evaluates the effectiveness of a set of suggestions with respect to their impact in diversifying the ranking for the initial query, as opposed to evaluating each suggestion individually. The only exception is the adhoc search scenario that considers only the top ranked suggestion ($k = 1$) for evaluation (e.g., for automatically reformulating the user's original query), which exhibits a low reuse of the TREC Web track assessments, with an average fraction of judged and relevant documents of 0.0344 and 0.0208, respectively. The evaluation in this specific scenario could be made more robust by incorporating alternative evaluation methodologies that take into account assessment sparsity [9,10], and by conducting additional relevance assessments, e.g., through crowdsourcing [1]. Alternatively, the effectiveness of a set of suggestions could be evaluated based upon their combined ability to improve the adhoc performance of the original query, in a similar fashion to our conducted diversity evaluation, as defined by Equation (3). This could be achieved either by diversifying the initial ranking [38], or by simply enriching it with results for different suggestions [41]. We leave these investigations for future work.

7 Conclusions

We have proposed a learning to rank approach for the query suggestion problem. Our approach represents candidate suggestions as structured virtual documents comprising terms from related queries with common clicks, in addition to those from common sessions, as proposed by previous research. Besides helping overcome query sparsity, this enriched representation enables multiple query-dependent features to be computed for each candidate suggestion, by matching the input query terms against the terms of related queries in the suggestion representation. We have also proposed several query-independent features specifically targeted at identifying quality suggestions. Finally, we have integrated all these features in order to automatically learn effective models for ranking candidate suggestions in response to a user's query.

To evaluate our proposed approach, we have introduced an evaluation framework that directly leverages relevance assessments from existing web search evaluation campaigns, hence requiring no extra assessment efforts, while being demonstrably robust to missing assessments. We have deployed this framework for quantitatively evaluating the effectiveness of query suggestions for two practical search scenarios, namely, to provide effective alternatives to the initial query, or to help diversify the results for this query. Under this framework, we have contrasted our learning to rank approach to a state-of-

the-art query suggestion baseline from the literature, as well as to query suggestions provided by a commercial web search engine. The results show that our approach significantly improves upon the baseline query suggestion mechanism, with a competitive performance compared to the web search engine's provided suggestions. This is a remarkable achievement, given that commercial search engines arguably use much larger query logs than the one-month log snapshot available to our approach.

In the future, we plan to further improve our approach by investigating more quality features, as well as by enhancing its underlying structured virtual document representation with refined estimations of the relationship between a candidate suggestion and its related queries. In addition, we plan to investigate approaches for explicitly diversifying the ranking of suggestions, in the hope of enabling an even more effective diversification of search results.

Acknowledgements

We are grateful to the feedback provided by the anonymous reviewers, particularly the reviewer who inspired the analysis presented in Section 6.4 on the robustness of our query suggestion evaluation framework with respect to missing document relevance assessments.

References

1. O. Alonso, D. E. Rose, and B. Stewart. Crowdsourcing for relevance evaluation. *SIGIR Forum*, 42(2):9–15, 2008.
2. G. Amati. *Probabilistic Models for Information Retrieval based on Divergence from Randomness*. PhD thesis, Univ. of Glasgow, 2003.
3. G. Amati, E. Ambrosi, M. Bianchi, C. Gaibisso, and G. Gambosi. FUB, IASI-CNR and University of Tor Vergata at TREC 2007 Blog track. In *Proc. of TREC*, 2007.
4. R. A. Baeza-Yates, C. A. Hurtado, and M. Mendoza. Query recommendation using query logs in search engines. In *Proc. of ClustWeb at EDBT*, pages 588–596, 2004.
5. P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna. The query-flow graph: model and applications. In *Proc. of CIKM*, pages 609–618, 2008.
6. P. Boldi, F. Bonchi, C. Castillo, D. Donato, and S. Vigna. Query suggestions using query-flow graphs. In *Proc. of WSCD at WSDM*, pages 56–63, 2009.
7. D. Broccolo, L. Marcon, F. M. Nardini, R. Perego, and F. Silvestri. Generating suggestions for queries in the long tail with an inverted index. *Inf. Process. Manage.*, 48(2):326–339, 2012.
8. C. J. C. Burges. From RankNet to LambdaRank to LambdaMART: an overview. Technical Report MSR-TR-2010-82, Microsoft Research, 2010.
9. B. Carterette, J. Allan, and R. Sitaraman. Minimal test collections for retrieval evaluation. In *Proc. of SIGIR*, pages 268–275, 2006.
10. B. Carterette, V. Pavlu, E. Kanoulas, J. A. Aslam, and J. Allan. If I dad a million queries. In *Proc. of ECIR*, pages 288–300. Springer, 2009.
11. O. Chapelle and Y. Chang. Yahoo! learning to rank challenge overview. *J. Mach. Learn. Res.*, 14:1–24, 2011.
12. O. Chapelle, D. Metlzer, Y. Zhang, and P. Grinspan. Expected reciprocal rank for graded relevance. In *Proc. of CIKM*, pages 621–630, 2009.
13. C. L. A. Clarke, N. Craswell, and I. Soboroff. Overview of the TREC 2009 Web track. In *Proc. of TREC*, 2009.

14. C. L. A. Clarke, N. Craswell, I. Soboroff, and A. Ashkan. A comparative analysis of cascade measures for novelty and diversity. In *Proc. of WSDM*, pages 75–84, 2011.
15. C. L. A. Clarke, N. Craswell, I. Soboroff, and G. V. Cormack. Overview of the TREC 2010 Web track. In *Proc. of TREC*, 2010.
16. C. L. A. Clarke, N. Craswell, I. Soboroff, and E. M. Voorhees. Overview of the TREC 2011 Web track. In *Proc. of TREC*, 2011.
17. C. L. A. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proc. of SIGIR*, pages 659–666, 2008.
18. C. L. A. Clarke, M. Kolla, and O. Vechtomova. An effectiveness measure for ambiguous and underspecified queries. In *Proc. of ICTIR*, pages 188–199, 2009.
19. S. Cucerzan and R. W. White. Query suggestion based on user landing pages. In *Proc. of SIGIR*, pages 875–876. ACM, 2007.
20. V. Dang, M. Bendersky, and W. B. Croft. Learning to rank query reformulations. In *Proc. of SIGIR*, pages 807–808. ACM, 2010.
21. J. Dean. Challenges in building large-scale information retrieval systems: invited talk. In *Proc. of WSDM*, page 1. ACM, 2009.
22. D. Downey, S. Dumais, and E. Horvitz. Heads and tails: studies of web search with common and rare queries. In *Proc. of SIGIR*, pages 847–848, 2007.
23. B. M. Fonseca, P. B. Golgher, E. S. De Moura, B. Pôssas, and N. Ziviani. Discovering search engine related queries using association rules. *J. Web Eng.*, 2:215–227, 2003.
24. Y. Ganjisaffar, R. Caruana, and C. Lopes. Bagging gradient-boosted trees for high precision, low variance ranking models. In *Proc. of SIGIR*, pages 85–94, Beijing, China, 2011.
25. C. Hauff, D. Kelly, and L. Azzopardi. A comparison of user and system query performance predictions. In *Proc. of CIKM*, pages 979–988, 2010.
26. B. J. Jansen, A. Spink, J. Bateman, and T. Saracevic. Real life information retrieval: a study of user queries on the web. *SIGIR Forum*, 32(1):5–17, 1998.
27. K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.
28. R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *Proc. of WWW*, pages 387–396, 2006.
29. T.-Y. Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331, 2009.
30. Q. Mei, D. Zhou, and K. Church. Query suggestion using hitting time. In *Proc. of CIKM*, pages 469–478, 2008.
31. D. Metzler. Automatic feature selection in the Markov random field model for information retrieval. In *Proc. of CIKM*, pages 253–262, 2007.
32. D. Metzler and W. B. Croft. A Markov random field model for term dependencies. In *Proc. of SIGIR*, pages 472–479, 2005.
33. I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, and C. Lioma. Terrier: a high performance and scalable information retrieval platform. In *Proc. of OSIR at SIGIR*, 2006.
34. J. Peng, C. Macdonald, B. He, V. Plachouras, and I. Ounis. Incorporating term dependency in the DFR framework. In *Proc. of SIGIR*. ACM Press, 2007.
35. T. Qin, T.-Y. Liu, J. Xu, and H. Li. LETOR: a benchmark collection for research on learning to rank for information retrieval. *Inf. Retr.*, 13(4):347–374, 2009.
36. S. Robertson. On the optimisation of evaluation metrics. In *Proc. of LR4IR at SIGIR*, 2008.
37. S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *Proc. of TREC*, 1994.
38. R. L. T. Santos, C. Macdonald, and I. Ounis. Exploiting query reformulations for web search result diversification. In *Proc. of WWW*, pages 881–890, 2010.
39. R. L. T. Santos, C. Macdonald, and I. Ounis. How diverse are web search results? In *Proc. of SIGIR*, pages 1187–1188, 2011.
40. R. L. T. Santos, C. Macdonald, and I. Ounis. Intent-aware search result diversification. In *Proc. of SIGIR*, pages 595–604, 2011.
41. D. Sheldon, M. Shokouhi, M. Szummer, and N. Craswell. LambdaMerge: merging the results of query reformulations. In *Proc. of WSDM*, pages 795–804, 2011.

42. F. Silvestri. Mining query logs: turning search usage data into knowledge. *Found. Trends Inf. Retr.*, 4(1–2):1–174, 2010.
43. R. Song, Z. Luo, J.-Y. Nie, Y. Yu, and H.-W. Hon. Identification of ambiguous queries in web search. *Inf. Process. Manage.*, 45(2):216–229, 2009.
44. Y. Song, D. Zhou, and L. wei He. Post-ranking query suggestion by diversifying search results. In *Proc. of SIGIR*, pages 815–824, Beijing, China, 2011.
45. K. Spärck-Jones, S. E. Robertson, and M. Sanderson. Ambiguous requests: Implications for retrieval tests, systems and theories. *SIGIR Forum*, 41(2):8–17, 2007.
46. I. Szpektor, A. Gionis, and Y. Maarek. Improving recommendation for long-tail queries via templates. In *Proc. of WWW*, pages 47–56, 2011.
47. X. Wang and C. Zhai. Mining term association patterns from search logs for effective query reformulation. In *Proc. of CIKM*, pages 479–488, 2008.
48. H. Zaragoza, N. Craswell, M. J. Taylor, S. Saria, and S. E. Robertson. Microsoft Cambridge at TREC 13: Web and Hard tracks. In *Proc. of TREC*, 2004.
49. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proc. of SIGIR*, pages 334–342, 2001.
50. Z. Zhang and O. Nasraoui. Mining search engine query logs for query recommendation. In *Proc. of WWW*, pages 1039–1040, 2006.