Lexical Query Modeling in Session Search

Christophe Van Gysel
cvangysel@uva.nl

Evangelos Kanoulas
e.kanoulas@uva.nl

Maarten de Rijke
derijke@uva.nl

University of Amsterdam, Amsterdam, The Netherlands

ABSTRACT

Lexical query modeling has been the leading paradigm for session search. In this paper, we analyze TREC session query logs and compare the performance of different lexical matching approaches for session search. Naive methods based on term frequency weighing perform on par with specialized session models. In addition, we investigate the viability of lexical query models in the setting of session search. We give important insights into the potential and limitations of lexical query modeling for session search and propose future directions for the field of session search.

1. INTRODUCTION

Many complex information seeking tasks, such as planning a trip or buying a car, cannot sufficiently be expressed in a single query [7]. These multi-faceted tasks are exploratory, comprehensive, survey-like or comparative in nature [14] and require multiple search iterations to be adequately answered [8]. Donato et al. [5] note that 10% of the user sessions (more than 25% of query volume) consists of such complex information needs.

The TREC Session Track [15] created an environment for researchers "to test whether systems can improve their performance for a given query by using previous queries and user interactions with the retrieval system." The track's existence led to an increasing number of methods aimed at improving session search. Yang et al. [16] introduce the Query Change Model (QCM), which uses lexical editing changes between consecutive queries in addition to query terms occurring in previously retrieved documents, to improve session search. They heuristically construct a lexicon-based query model for every query in a session. Query models are then linearly combined for every document, based on query recency [16] or document satisfaction [3, 10], into a session-wide lexical query model. However, there has been a clear trend towards the use of supervised learning [3, 12, 16] and external data sources [6, 11]. Guan et al. [6] perform lexical query expansion by adding higher-order n-grams to queries by mining document snippets. In addition, they expand query representations by including anchor texts to previously top-ranked documents in the session. Carterette et al. [3] expand document representations by including incoming anchor texts. Luo et al. [12] introduce a linear point-wise learning-to-rank model that predicts relevance given a document and query change features. They incorporate document-independent session features in their ranker.

The use of machine-learned ranking and the expansion of query and document representations is meant to address a specific instance of a wider problem in information retrieval, namely the query document mismatch [9]. In this paper, we analyze the session query logs made available by TREC and compare the performance of different lexical query modeling approaches for session search, taking into account session length.[1] In addition, we investigate the viability of lexical query models in a session search setting.

The main purpose of this paper is to investigate the potential of lexical methods in session search and provide foundations for future research. We ask the following questions: (1) Increasingly complex methods for session search are being developed, but how do naive methods perform? (2) How well can lexical methods perform? (3) Can we solve the session search task using lexical matching only?

2. LEXICAL MATCHING FOR SESSIONS

We define a search session s as a sequence of n interactions (q_i, r_i) between user and search engine, where q_i denotes a user-issued query consisting of $|q_i|$ terms $t_{i,1}, \ldots, t_{i,|q_i|}$ and r_i denotes a result page consisting of $|r_i|$ documents $r_{i,1}, \ldots, r_{i,|r_i|}$ returned by the search engine (also referred to as SERP). The goal, then, is to return a SERP r_{n+1} given a query q_{n+1} and the session history that maximizes the user's utility function.

In this work, we formalize session search by modeling an observed session s as a query model parameterized by $\theta^s = \{\theta^s_1, \ldots, \theta^s_{|V|}\}$, where θ^s_i denotes the weight associated with term $t_i \in V$ (specified below). Documents d_j are then ranked in decreasing order of

$$\log P(d_j \mid s) = \sum_{k=1}^{|V|} \theta^s_k \log \theta^{d_j}_k,$$

where θ^{d_j} is a lexical model of document d_j, which can be a language model (LM), a vector space model or a specialized model using hand-engineered features. Query model θ^s is a function of the query models of the interactions i in the session, θ^{s_i} (e.g., for a uniform aggregation scheme, $\theta^s = \sum_i \theta^{s_i}$). Existing session search methods [6, 16] can be expressed in this formalism as follows:

Term frequency (TF) Terms in a query are weighted according to their frequency in the query (i.e., $\theta^{s_i}_k$ becomes the frequency

[1] An open-source implementation of our testbed for evaluating session search is available at https://github.com/cvangysel/sesh.

ICTIR '16, September 12 - 16, 2016, Newark, DE, USA

ISBN 978-1-4503-4497-5/16/09...$15.00

DOI: http://dx.doi.org/10.1145/2970398.2970422

of term t_k in q_i). Queries q_i that are part of the same session s are then aggregated uniformly for a subset of queries. In this work, we consider the following subsets: the *first query*, the *last query* and the *concatenation of all queries* in a session. Using the last query corresponds to the official baseline of the TREC Session track [3].

Nugget Nugget [6] is a method for effective structured query formulation for session search. Queries q_i, part of session s, are expanded using higher order n-grams occurring in both q_i and snippets of the top-k documents in the previous interaction, $r_{i-1,1}, \ldots, r_{i-1,k}$. This effectively expands the vocabulary by additionally considering n-grams next to unigram terms. The query models of individual queries in the session are then aggregated using one of the aggregation schemes. Nugget is primarily targeted at resolving the query-document mismatch by incorporating structure and external data and does not model query transitions. The method can be extended to include external evidence by expanding θ^s to include anchor texts pointing to (clicked) documents in previous SERPs.

Query Change Model (QCM) QCM [16] uses syntactic editing changes between consecutive queries in addition to query changes and previous SERPs to enhance session search. In QCM [16, Section 6.3], document model θ^d is provided by a language model with Dirichlet smoothing and the query model at interaction i, θ^{s_i}, in session s is given by

$$\theta_k^{s_i} = \begin{cases} 1 + \alpha(1 - P(t_k \mid r_{i-1,1})), & t_k \in q_{\text{theme}} \\ 1 - \beta P(t_k \mid r_{i-1,1}), & t_k \in +\Delta q \wedge t_k \in r_{i-1,1} \\ 1 + \epsilon \,\text{idf}(t_k), & t_k \in +\Delta q \wedge t_k \notin r_{i-1,1} \\ -\delta P(t_k \mid r_{i-1,1}), & t_k \in -\Delta q, \end{cases}$$

where q_{theme} are the session's theme terms, $+\Delta q$ ($-\Delta q$, resp.) are the added (removed) terms, $P(t_k \mid r_{i-1,1})$ denotes the probability of t_k occurring in SAT clicks, $\text{idf}(t_k)$ is the inverse document frequency of term t_k and α, β, ϵ, δ are parameters. The θ^{s_i} are then aggregated into θ^s using one of the aggregation schemes, such as the uniform aggregation scheme (i.e., the sum of the θ^{s_i}).

In §4, we analyze the methods listed above in terms of their ability to handle sessions of different lengths and contextual history.

3. EXPERIMENTS

3.1 Benchmarks

We evaluate the lexical query modeling methods listed in §2 on the session search task (G1) of the TREC Session track from 2011 to 2014 [15]. We report performance on each track edition independently and on the track aggregate. Given a query, the task is to improve retrieval performance by using previous queries and user interactions with the retrieval system. To accomplish this, we first retrieve the 2,000 most relevant documents for the given query and then re-rank these documents using the methods described in §2. We use the "Category B" subsets of ClueWeb09 (2011/2012) and ClueWeb12 (2013/2014) as document collections. Both collections consist of approximately 50 million documents. Spam documents are removed before indexing by filtering out documents with scores (GroupX and Fusion, respectively) below 70 [4]. Table 1 shows an overview of the benchmarks and document collections.

3.2 Evaluation measures

To measure retrieval effectiveness, we report Normalized Discounted Cumulative Gain at rank 10 (NDCG@10) in addition to Mean Reciprocal Rank (MRR). The relevance judgments of the tracks were converted from topic-centric to session-centric according to the mappings provided by the track organizers.[2] Evaluation measures are then computed using TREC's official evaluation tool, `trec_eval`.[3]

3.3 Systems under comparison

We compare the lexical query model methods outlined in §2. All methods compute weights for lexical entities (e.g., unigram terms) on a per-session basis, construct a structured Indri query [13] and query the document collection using `pyndri`.[4] For fair comparison, we use Indri's default smoothing configuration (i.e., Dirichlet smoothing with $\mu = 2500$) and uniform query aggregation for all methods (different from the smoothing used for QCM in [16]). This allows us to separate query aggregation techniques from query modeling approaches in the case of session search.

For Nugget, we use the default parameter configuration ($k_{\text{snippet}} = 10$, $\theta = 0.97$, $k_{\text{anchor}} = 5$ and $\beta = 0.1$), using the strict expansion method. We report the performance of Nugget without the use of external resources (RL2), with anchor texts (RL3) and with click data (RL4). For QCM, we use the parameter configuration as described in [12, 16]: $\alpha = 2.2$, $\beta = 1.8$, $\epsilon = 0.07$ and $\delta = 0.4$.

In addition to the methods above, we report the performance of an oracle that always ranks in decreasing order of ground-truth relevance. This oracle will give us an upper-bound on the achievable ranking performance.

3.4 Ideal lexical term weighting

We investigate the maximally achievable performance by weighting query terms. Inspired by Bendersky et al. [1], we optimize NDCG@10 for every session using a grid search over the term weight space. We sweep the weight of every term between -1.0 and 1.0 (inclusive) with increments of 0.1, resulting in a total of 21 weight assignments per term. Due to the exponential time complexity of the grid search, we limit our analysis to the 230 sessions with 7 unique query terms or less (see Table 1). This experiment will tell us the maximally achievable retrieval performance in session search by the re-weighting lexical terms only.

4. RESULTS & DISCUSSION

In this section, we report and discuss our experimental results. Of special interest to us are the methods that perform lexical matching based on a user's queries in a single session: QCM, Nugget (RL2) and the three variants of TF. Table 2 shows the methods' performance on the TREC Session track editions from 2011 to 2014. No single method consistently outperforms the other methods. Interestingly enough, the methods based on term frequency (TF) perform quite competitively compared to the specialized session search methods (Nugget and QCM). In addition, the TF variant using all queries in a session even outperforms Nugget (RL2) on the 2011 and 2014 editions and QCM on nearly all editions. Using the concatenation of all queries in a session, while being an obvious baseline, has not received much attention in recent literature or by TREC [15]. In addition, note that the best-performing (unsupervised) TF method achieves better results than the supervised method of Luo et al. [12] on the 2012 and 2013 tracks. Fig. 1 depicts the boxplot of the NDCG@10 distribution over all track editions (2011–2014). The term frequency approach using all queries

[2]We take into account the mapping between judgments and actual relevance grades for the 2012 edition.

[3]https://github.com/usnistgov/trec_eval

[4]https://github.com/cvangysel/pyndri

Table 1: Overview of 2011, 2012, 2013 and 2014 TREC session tracks. For the 2014 track, we report the total number of sessions in addition to those sessions with judgments. We report the mean and standard deviation where appropriate; M denotes the median.

	2011	2012	2013	2014
Sessions				
Sessions	76	98	87	100 (1,021 total)
Queries per session	3.68 ± 1.79; M=3.00	3.03 ± 1.57; M=2.00	5.08 ± 3.60; M=4.00	4.34 ± 2.22; M=4.00
Unique terms per session	7.01 ± 3.28; M=6.50	5.76 ± 2.95; M=5.00	8.86 ± 4.38; M=8.00	7.79 ± 4.08; M=7.00
Topics				
Session per topic	1.23 ± 0.46; M= 1.00	2.04 ± 0.98; M= 2.00	2.18 ± 0.93; M= 2.00	20.95 ± 4.81; M= 21.00
Document judgments per topic	313.11 ± 114.63; M=292.00	372.10 ± 162.63; M=336.50	268.00 ± 116.86; M=247.00	332.33 ± 149.03; M=322.00
Collection				
Documents	21,258,800		15,702,181	
Document length	1,096.18 ± 1,502.45		649.07 ± 1,635.29	
Terms	3.40×10^7 (2.33×10^{10} total)		2.36×10^7 (1.02×10^{10} total)	
Spam scores	GroupX		Fusion	

Table 2: Overview of experimental results on 2011–2014 TREC Session tracks of the TF, Nugget and QCM methods (see §2). The ground-truth oracle shows the ideal performance (§3.3).

	2011		2012		2013		2014	
	NDCG@10	MRR	NDCG@10	MRR	NDCG@10	MRR	NDCG@10	MRR
Ground-truth oracle	0.777	0.868	0.695	0.865	0.517	0.920	0.410	0.800
TF (first query)	0.371	0.568	0.302	0.523	0.121	0.379	0.120	0.336
TF (last query)	0.358	0.598	0.316	0.586	0.133	0.358	0.156	0.458
TF (all queries)	**0.448**	**0.685**	0.348	0.604	0.162	0.477	**0.174**	**0.478**
Nugget (RL2)	0.437	0.677	0.352	0.609	**0.163**	0.488	0.173	0.476
Nugget (RL3)	0.442	0.678	**0.360**	**0.619**	0.162	**0.488**	0.172	0.477
Nugget (RL4)	0.437	0.677	0.352	0.609	0.163	0.488	0.173	0.476
QCM	0.440	0.661	0.342	0.575	0.160	0.484	0.162	0.450

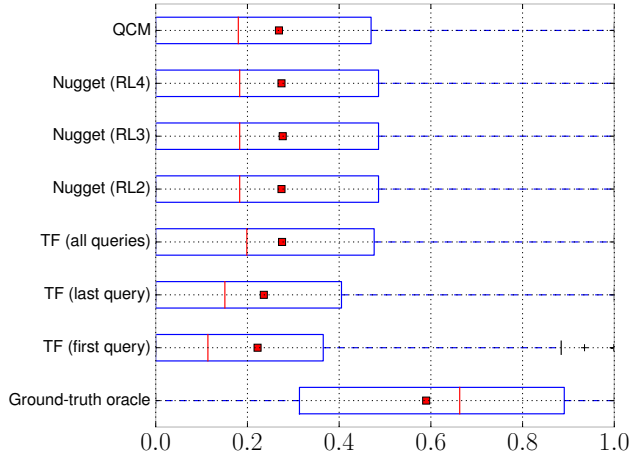
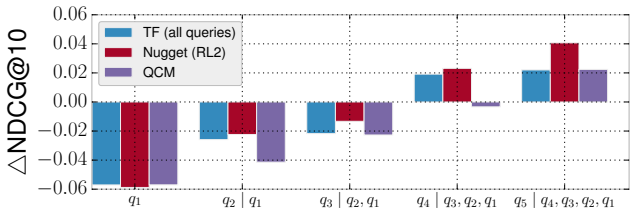


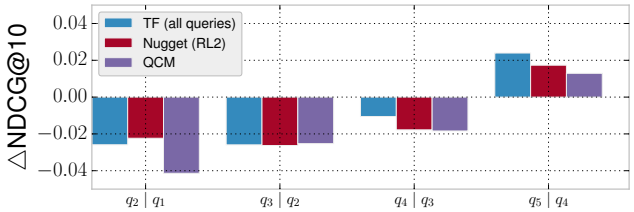
Figure 1: Box plot of NDCG@10 on all sessions of the TREC Session track (2011–2014). The box depicts the first, second (median) and third quartiles. The whiskers are located at 1.5 times the interquartile range on both sides of the box. The square and crosses depict the average and outliers respectively.

achieves the highest mean/median overall. Given this peculiar finding, where a generic retrieval model performs better than specialized session search models, we continue with an analysis of the TREC Session search logs.

In Fig. 2 we investigate the effect of varying session lengths in the session logs. The distribution of session lengths is shown in the top row of Fig. 2. For the 2011–2013 track editions, most sessions consisted of only two queries. The mode of the 2014 edition lies at 5 queries per session. If we examine the performance of the methods on a per-session length basis, we observe that the TF methods perform well for short sessions. This does not come as a surprise, as for these sessions there is only a limited history that specialized methods can use. However, the TF method using the concatenation of all queries still performs competitively for longer sessions. This can be explained by the fact that as queries are aggregated over time, a better representation of the user's information need is created. This aggregated representation naturally emphasizes important *theme terms* of the session, which is a key component in the QCM [16].

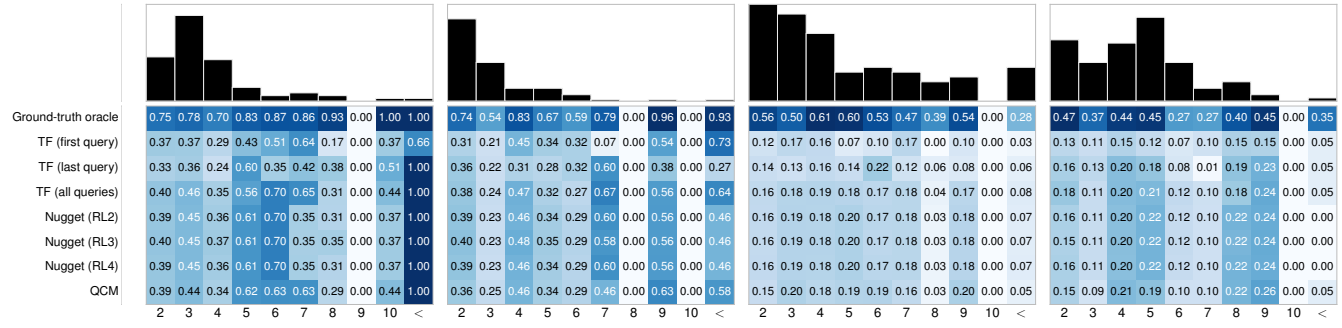


(a) Full history of session



(b) Previous query in session only

Figure 3: Difference in NDCG@10 with the official TREC baseline (TF using the last query only) of 5-query sessions (45 instances) with different history configurations for the 2011–2014 TREC Session tracks.



(a) 2011 **(b) 2012** **(c) 2013** **(d) 2014**

Figure 2: The top row depicts the distribution of session lengths for the 2011–2014 TREC Session tracks, while the bottom row shows the performance of the TF, Nugget and QCM models for different session lengths.

Table 3: NDCG@10 for TF weighting (§2), ideal term weighting (§3.4) and the ground-truth oracle (§3.3).

	2011	2012	2013	2014
TF (all queries)	0.391	0.333	0.179	0.183
Ideal term weighing	0.589	0.528	0.361	0.296
Ground-truth oracle	0.716	0.682	0.593	0.453

How do these methods perform as the search session progresses? Fig. 3 shows the performance of sessions of length five after every user interaction, when using all queries in a session (Fig. 3a) and when using only the previous query (Fig. 3b). We can see that NDCG@10 increases as the session progresses for all methods. Beyond half of the session, the session search methods outperform retrieving according to the last query in the session. We see that, for longer sessions, specialized methods (Nugget, QCM) outperform generic term frequency models. This comes as no surprise. Bennett et al. [2] note that users tend to reformulate and adapt their information needs based on observed results and this is essentially the observation upon which QCM builds.

Fig. 1 and Table 2 reveal a large NDCG@10 gap between the compared methods and the ground-truth oracle. How can we bridge this gap? Table 3 shows a comparison between frequency-based term weighting, the ideal term weighting (§3.4) and the ground-truth oracle (§3.3) for all sessions consisting of 7 unique terms or less (§3.4). Two important observations. There is still plenty of room for improvement using lexical query modeling only. Relatively speaking, around half of the gap between weighting according to term frequency and the ground-truth can be bridged by predicting better term weights. However, the other half of the performance gap cannot be bridged using lexical matching only, but instead requires a notion of semantic matching [9].

5. CONCLUSIONS

We have shown that naive frequency-based term weighting methods perform on par with specialized session search methods on the TREC Session track (2011–2014).[5] This is due to the fact that shorter sessions are more prominent in the session query logs. On longer sessions, specialized models are able to exploit session history more effectively. Future work should focus on creating benchmarks consisting of longer sessions with complex information needs. Perhaps more importantly, we have looked at the viability of lexical query matching in session search. There is still much room for improvement by re-weighting query terms. However, the query/document mismatch is prevalent in session search and methods restricted to lexical query modeling face a very strict performance ceiling. Future work should focus on better lexical query models for session search, in addition to semantic matching and tracking the dynamics of contextualized semantics in search.

[5]An open-source implementation of our testbed for evaluating session search is available at https://github.com/cvangysel/sesh.

Acknowledgments This work was supported by the Google Faculty Research Award and the Bloomberg Research Grant programs. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors. The authors would like to thank Daan Odijk, David Graus and the anonymous reviewers for their valuable comments and suggestions.

REFERENCES

[1] M. Bendersky, D. Metzler, and W. B. Croft. Effective query formulation with multiple information sources. In *SIGIR*, pages 443–452. ACM, 2012.
[2] P. N. Bennett, R. W. White, W. Chu, S. T. Dumais, P. Bailey, F. Borisyuk, and X. Cui. Modeling the impact of short- and long-term behavior on search personalization. In *SIGIR*, pages 185–194. ACM, 2012.
[3] B. Carterette, E. Kanoulas, M. M. Hall, and P. D. Clough. Overview of the trec 2014 session track. In *TREC*, 2014.
[4] G. V. Cormack, M. D. Smucker, and C. L. Clarke. Efficient and effective spam filtering and re-ranking for large web datasets. *Information retrieval*, 14(5):441–465, 2011.
[5] D. Donato, F. Bonchi, T. Chi, and Y. Maarek. Do you want to take notes?: identifying research missions in yahoo! search pad. In *WWW*, pages 321–330. ACM, 2010.
[6] D. Guan, H. Yang, and N. Goharian. Effective structured query formulation for session search. Techn. report, 2012.
[7] A. Hassan, R. W. White, S. T. Dumais, and Y.-M. Wang. Struggling or exploring?: disambiguating long search sessions. In *WSDM*, pages 53–62. ACM, 2014.
[8] A. Kotov, P. N. Bennett, R. W. White, S. T. Dumais, and J. Teevan. Modeling and analysis of cross-session search tasks. In *SIGIR*, pages 5–14. ACM, 2011.
[9] H. Li and J. Xu. Semantic matching in search. *Found. & Tr. in Information Retrieval*, 7(5):343–469, June 2014.
[10] J. Luo, X. Dong, and H. Yang. Modeling rich interactions in session search - georgetown university at trec 2014 session track. Techn. report, 2014.
[11] J. Luo, S. Zhang, and H. Yang. Win-win search: Dual-agent stochastic game in session search. In *SIGIR*, pages 587–596. ACM, 2014.
[12] J. Luo, X. Dong, and H. Yang. Session search by direct policy learning. In *ICTIR*, pages 261–270. ACM, 2015.
[13] D. Metzler and W. B. Croft. Combining the language model and inference network approaches to retrieval. *IPM*, 40(5):735–750, 2004.
[14] K. Raman, P. N. Bennett, and K. Collins-Thompson. Toward whole-session relevance: exploring intrinsic diversity in web search. In *SIGIR*, pages 463–472. ACM, 2013.
[15] TREC. Session Track, 2009–2014.
[16] H. Yang, D. Guan, and S. Zhang. The query change model: Modeling session search as a markov decision process. *TOIS*, 33(4): 20:1–20:33, May 2015.