RGU-ISTI-Essex at TREC 2011 Session Track

Ibrahim Adeyanju[1], Franco Maria Nardini[2], M-Dyaa Albakour[3], Dawei Song[1], and Udo Kruschwitz[3]

[1] IDEAS Research Institute, The Robert Gordon University,
Aberdeen, AB25 1HG, Scotland, UK
[i.adeyanju|d.song]@.rgu.ac.uk

[2] Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo",
Consiglio Nazionale delle Ricerche,
Via G. Moruzzi, 1, 56124, Pisa, Italy
francomaria.nardini@isti.cnr.it

[3] School of Computer Science and Electronic Engineering, University of Essex,
Wivenhoe Park, Colchester, CO4 3SQ, UK
[malbak|udo]@essex.ac.uk

Abstract. Mining query recommendation from query logs has attracted a lot of attention in recent years. We propose to use query recommendations extracted from the logs of a web search engine to solve the session track tasks. The runs are obtained by using the Search Shortcuts recommender system. The Search Shortcuts technique uses an inverted index and the concept of "successful sessions" present in a web search engine's query log to produce effective recommendations for both frequent and rare/unseen queries. We adapt the above technique as a query expansion tool and use it to expand the given queries for Session Track at TREC 2011. The expansion is generated by using a method which aims to consider all past queries in the session. The expansion terms obtained are then used to build a global, uniformly weighted, representation of the user session (RL2). Furthermore, the expansion terms are then combined with a ranked list of results in order to boost terms appearing more frequently in the final results lists (RL3). Finally, we also integrate dwell times and the weighting method obtained taking both result lists and clicks into account for assigning weights to the terms to expand the final query of the session. In addition to that, we submitted a baseline run. It is based on the observation that using the term "wikipedia" to expand the query resulted in a better retrieval performance for the tasks at last year's session track at TREC 2010.

Keywords Search shortcuts, Session track, TREC 2011, Query logs

1 Introduction

The Session Track, introduced at the Text REtrieval Conference (TREC) 2010, aims to evaluate the ability of search engines to use previous user interactions in order to provide better results for subsequent queries in a user session thereby

guiding a user to find relevant information faster. Last year participants were given query sessions containing only two queries and no interaction data. This year, the session track provided more interactive data to the participants. Query sessions were collected from real users and contained a variable number of queries and interaction data such as the documents displayed to the user, the clicked documents and dwelling times.

Session track 2011 provided another opportunity for us to propose a couple of techniques to improve the retrieval performance over user sessions. We submitted one baseline run and two other runs that make use of the Search Shortcuts (SS) [2] query recommendation technique. The two runs (apart from baseline) exploit the effectiveness of SS in producing query recommendations. SS is a query recommendation technique that make use of *successful* sessions (i.e., sessions ending with at least a click on the last query) present in a query log to produce suggestions. The technique is not only generally efficient and very effective but also works well for queries in the long tail of the distribution [2].

The SS technique is able to use "session" information to devise recommendations for a given query. This is practically due to the organization of the knowledge model which makes use of an inverted index for computing recommendations. For each current query of the given sessions, we produce its relative recommendations. Furthermore, we use them to expand the current query with the terms composing them. We study different ways of expand the query with the terms obtained. First, we combine the current query with the terms produced by the SS recommender system by uniformly weighting them ($RL2$). Second, we use the ranked lists of results provided for each session as a way to boost terms that appears more frequently in the result lists ($RL3$). The last list ($RL4$) of the two runs makes use of dwelling time and available clicks on the result lists for assigning weights to the terms to expand the current query of the session.

The rest of the paper is structured as follows. Section 2 gives a brief description of the Session Track tasks for this year. We introduce the methodology used in building our runs in Section 3 and discuss the dataset and the resources used in our runs in Section 4. Details of our experiments and runs submitted to TREC appear in Section 5 with the evaluation results are discussed in Section 6. Finally, a brief conclusion is given in Section 7.

2 Session Track 2011 Tasks

The main difference in this year task compared to the last year is that more interactive data is provided to the participants. Participants are provided with a set of query sessions. Each session consists of the current query q_m and the user interactive data prior to the current query. These data includes:

(a) the set of past queries in the session q_1, q_2, ..., q_{m-1},
(b) the ranked list of URLs for each past query,
(c) the set of clicked URLs/snippets and the time spent by the user reading the corresponding to each clicked url webpage.

Participants are tasked to run their retrieval system over the current query using the four criteria enumerated below.

1. ignoring the session data prior to the current query ($RL1$),
2. considering only the item (a) above, i.e. the queries prior to the current query ($RL2$),
3. considering only the items (a) and (b) above, i.e. the queries prior to the current along with the ranked lists of URLs and the corresponding web pages ($RL3$),
4. considering only the items (a), (b) and (c) above, i.e the queries prior to the current, the ranked lists of URLs and the corresponding web pages and the clicked URLs and the time spent on the corresponding web pages ($RL4$).

3 The Search Shortcuts Recommender System

A search session is an interactive process where users continuously refine their search query in order to better specify their information need. Sometimes, the successful query is not known in advance, but users might adopt concepts and terminologies on the basis of the results pages visited. The SS model we used was firstly proposed in [1] and then effectively designed in [2]. Let $\mathcal{U}$ be the set of users of a Web search engine whose activities are recorded in a query log QL, and $\mathcal{Q}$ be the set of queries in QL. We suppose QL is preprocessed by using some session splitting methods (e.g. [5, 8]) in order to extract query sessions, i.e., sequences of queries which are related to the same user search task. We say that a session σ is *successful* if and only if the user has clicked on at least one link shown in the result page returned by the search engine for the final query σ_n and *unsuccessful* otherwise.

The SS algorithm [2] works by efficiently computing similarities between partial user sessions (the one currently being performed) and historical successful sessions recorded in a query log. Final queries of most similar successful sessions are suggested to users as search shortcuts. Let σ' be the current session performed by the user, and let the sequence τ be the concatenation of all terms with possible repetitions appearing in $\sigma'_{t|}$, i.e. the head of length t of session σ'. Then, the algorithm computes the value of a scoring function $\delta(\tau, \sigma^s)$, which for each successful session measures the similarity between its queries and the set of terms τ. Intuitively, this similarity measures how much a previously seen session overlaps with the user need expressed so far (the concatenation of terms τ serves as a bag-of-words model of user need). Sessions are ranked according to δ scores and from the subset of the top ranked sessions, it is possible to suggest their final queries. It is obvious that depending on how the function δ is chosen the algorithm provides different recommendation. In the original paper, authors opted for δ to be the similarity computed as in the BM25 metrics [12]. An IR-like metric takes care of words that are discriminant in the context of the session to which we are comparing. BM25, and other IR-related metrics, have been designed specifically to account for that property in the context of query/documents similarity.

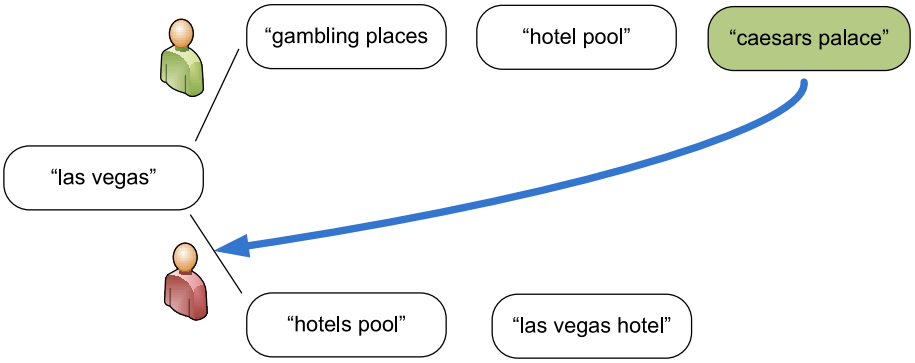

Fig. 1. Illustration of the Search Shortcuts method

Figure 3 illustrates how the search shortcuts are utilized for query recommendation. Here, a previous user whose final query was "caesars palace" had a successful session; that is, the user clicked on at least one of the results returned by the search engine for the final query. We can therefore reduce search/ refinement duration of a new user by suggesting the final query from the previous successful session with identical queries. As shown in the figure, rather than allowing the user to refine the initial query to "hotels pool" and "las vegas hotel", "caesars palace" is more likely to direct the user to a landing page with more relevant results thereby creating another successful session.

The idea described above is translated into the following process. For each unique *final query* q_f contained in successful sessions authors define a *virtual document* identified by its *title* and its *content*. The title, i.e., the identifier of the document, is exactly the query string q_f. The content of the virtual document is instead composed of all the terms that have appeared in queries of all the successful sessions ending with q_f. At the end of this procedure, a set of virtual documents is obtained; one for each distinct final query occurring in some successful sessions. All virtual documents are indexed with the preferred Information Retrieval system, and generating shortcuts for a given user session σ' is simply a matter of processing the query $\sigma'_{t|}$ over the inverted file indexing of such virtual documents. The processing of queries over inverted indexes is very fast and scalable, and these important characteristics are inherited by our query suggestion technique as well.

4 Experimental Setup

The ClueWeb09 dataset[1] is a web crawl of more than a billion pages that was first used in TREC 2009 Web track. The ClueWeb09 category B dataset is a

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

subset of the larger ClueWeb09 crawl and it consists of 50 million English pages. In this year Session Track's tasks, participants were permitted to use either one of the two datasets. An existing Indri[2] index of the ClueWeb09 dataset is already available and searchable via a public web service[3]. The web service enabled us to issue queries and retrieve the top documents returned by the search engine, thus removing the burden of indexing the data internally. The Indri search engine [7, 10] uses language modelling probabilities and supports query expansion.

The expanded representations of queries are obtained by using the Microsoft RFP 2006 query log which was preliminarily preprocessed by converting all queries to lower-case, and removing stop-words and punctuation/control characters. The queries in the log were then sorted by user and time-stamp, and segmented into sessions on the basis of a splitting algorithm which simply groups in the same session all the queries issued by the same users in a time span of 30 minutes [11]. Noisy sessions, likely performed by software robots, were removed. The remaining entries correspond to approximately nine million (9M) sessions.

5 Our Runs

Each group participating in the session track can submit a maximum of three different runs to TREC based on the tasks described in Section 2. Each submission file of the TREC Session Track 2011 consists of four different ranked lists, namely $RL1$, $RL2$, $RL3$, and $RL4$.

The current query q_m was processed to produce q'_m following these steps:

1. removing the following punctuation marks ( ) , ?
2. removing stop words from a common list of English stop words that do not fall within quoted text.
3. replacing quotes with the corresponding Indri syntax #1(<quoted text>) e.g. *"event planning"* becomes *#1(event planning)*
4. replacing site specification with the corresponding Indri format, e.g. "female winemakers site:.com.au" becomes "female winemakers com.url au.url"

The maximum number of returned documents in the list has been limited to 1000. We also used the Waterloo Spam Rankings[4] for the ClueWeb09 dataset to filter the spam documents from the returned ranked lists. We consider documents with scores of 70% or less as spam which is recommended by the creators of those rankings [4].

In all the runs, we generate $RL1$ by simply submitting a preprocessed version of the current query q'_m to the Indri index, i.e. $RL1$ is equivalent to $D_{q'_m}$. Generating $RL2$ this year is a similar task to generating $RL3$ in the previous year's task [6]. In our runs, we generate $RL2$ in different ways. The major challenge resides in generating $RL3$ and $RL4$. The sections below describe the techniques used for determining the other three runs of each submission, namely: $RL2$, $RL3$, $RL4$. All the runs are obtained by means of the Indri query language.

[2] http://lemurproject.org/indri.php
[3] http://boston.lti.cs.cmu.edu:8085/clueweb09/search/cataenglish/lemur.cgi
[4] http://durum0.uwaterloo.ca/clueweb09spam/

5.1 System 1 – Baseline (rguBase)

The first system we propose is the rguBase baseline. This baseline has been produced by expanding the current query of each session with the term "wikipedia". The rationale for this is that by doing so in Session Track 2010 [9], we obtain better retrieval performance. This submission file thus contains only two runs: $RL1$, and $RL2$ as we would like to assess only how the term "wikipedia" affect the overall retrieval performances of the given current query.

5.2 System 2 – Search Shortcuts (rguPisaSS)

The Search Shortcuts query recommender has been used as a query expansion technique. We build $RL2$ by using an expansion of the current query made by the terms composing the first three recommendations provided by the method and by uniformly weighting them after performing a stopword removal step. Furthermore, $RL3$ has been built starting by the first ten recommendations generated for each current query of the sessions provided. We use the terms composing them to produce an expanded query representation. In particular, after removing stopwords, we develop a term weighting scheme based on how many times any given expansion term appears in the snippets of the documents returned within the session. The final weight is thus computed by dividing the frequency of the single expansion term with the sum of the frequencies of the expansion terms over all the returned documents.

More formally, let E be the set of the expansion terms obtained by the recommendations produced by Search Shortcuts. Furthermore, let D be the set of documents returned for all the queries of the current user session. Let $f_D(t)$ be a function measuring the frequency of the term $t \in E$ in the set of the snippets returned for the documents in D. The expansion weights of the term t used within $RL3$ is thus derived using Equation 1.

$$w_t^{RL3} = \frac{f_D(t)}{\sum_{\forall x \in E} f_D(x)} \tag{1}$$

```
#weight(
0.7 #combine(event planning college)
0.3 #weight(0.16 #combine(college) 0.01 #combine(fashion) 0.36
#combine(event) 0.4 #combine(planning) 0.01 #combine(conference)
0.05 #combine(online) 0.01 #combine(management))
)
```

Fig. 2. An example of an expanded query with Search Shortcuts.

To illustrate the expansion process, Figure 2 shows a generated Indri query for session 2 ($RL3$). Note that for all the sessions and each ranked list we arbitrarily

chose the values 0.7 and 0.3 for both components of the expansion, i.e., the original query and the weighted expansion set of terms produced by the Search Shortcuts method.

RL4 has been produced starting from the weights obtained for *RL3* and by adding to the terms appearing in the snippets of the clicked documents a boosting factor. This boosting factor depends on the frequency (with repetitions) of the given term in the set of the clicked documents divided by the total frequency of the expansion terms that are present in the set of the clicked documents. More formally, let C be the set of the clicked documents for all the queries of the current user session. Clearly, $C \subseteq D$. w_t^{RL4} can be thus computed by applying Equation 2.

$$w_t^{RL4} = w_t^{RL3} + \left(\frac{f_C(t)}{\sum_{\forall x \in E} f_C(x)} \right) \tag{2}$$

Figure 3 compares the expansion weights for session 63 across *RL3* and *RL4*. It can be observed that the boosting factor increases the weights of some expansion terms while reducing the weights of others with respect to their RL3 weights after normalisation. Thus, in this example the RL4 weights of "court" and "judge" are boosted due to their frequency in the set of clicked documents.

Expansion term	**RL3 weight**	**RL4 weight**
court	0.7	0.85
judges	0.12	0.06
district	0.05	0.02
judge	0.02	0.05
united	0.01	0.01

Fig. 3. Sample expansion terms extracted for session 63: {FISA → judges on fisa court → 1990 FISA wiretap applications → judges FISA court → judges FISA court 2005}

5.3 System 3 – Search Shortcuts with Time (rguPisaSST)

In the rguPisaSST run, we build *RL2* by using an expansion of the current query made by the terms composing the first three recommendations provided by the method and by uniformly weighting them after performing a stopwords removal step. Furthermore, *RL3* has been built starting by the first ten recommendations generated for each current query of the sessions provided. We use the terms composing them to produce an expanded query representation. In particular, we remove stopwords and we develop a term weighting scheme based on how much any given expansion term appears in the snippets of the documents returned within the session. As in the previous case, *RL3* can be computed by applying Equation 1.

This run differs from the previous one on how we build the $RL4$ query. We produce $RL4$ by adding to the weights produced for $RL3$ a new weight obtained by exploiting the dwell time of each clicked document. The new weight measures how much a clicked document has been visualized by the user. We do this on a term basis. We thus select a candidate expansion term, we check if it is part of the snippet of one or more clicked documents and we compute its weight as a sum of its dwelling time within the session divided by the total dwelling time of all the documents within the session. We then sum the weights obtained to the weights referring to $RL3$. Finally, we normalize over all the expansion terms to obtain a set of weights that sum to one. More formally, let T be the set of documents that have been visualized for all the queries of the current user session. Clearly, $T \subseteq C$. In addition, let $f_T(t)$ be a function measuring the total dwelling time of the term $t \in E$. $f_T(t)$ works by computing the sum of the dwelling time of the documents in T for the term t (i.e. containing t in their snippets). Each $f_T(t)$ is then normalized by using the sum over all the terms $x \in E$ as shown in Equation 3.

$$w_t^{RL4} = w_t^{RL3} + \left(\frac{f_T(t)}{\sum_{\forall x \in E} f_T(x)} \right) \tag{3}$$

6 Results and Discussion

Tables 1 and 2 summarize the nDCG and nDCG@10 results for our three runs with significant test data as well as the maximum, median and minimum across all the session track participants. These results (for nDCG@10) are also charted without significance test in the graphs in Figures 4 and 5. This year, NIST assessors provided relevance assessments on two different criteria; thus, the results in Table 1 take into account the subtopics for all queries, whereas the results in Table 2 only reflect the current (last) query's subtopics.

Each of the table columns represent the normalised discounted cumulative gain (nDCG) for each result list submitted. Despite receiving a variety of eight relevance metrics which includes Expected Reciprocal Rank (ERR) [3], Average Precision [14] and Graded Average Precision (GAP) [13] among others, we will focus our results analysis on the nDCG results, as they relate to the established metrics from last year's TREC session track. The tables are split in two, the top half using the nDCG metric for all the documents submitted and the bottom showing the nDCG score using only the first ten returned documents (nDCG@10).

The arrows in the RL2, RL3 and RL4 columns represent the relative improvement or decline in the nDCG scores between the results lists for a given system (row), with a double arrow up ($\Uparrow$) or down ($\Downarrow$) indicating that a two tail t-test has supported the result as significantly better or worse. We use the horizontal double edge arrow ($\leftrightarrow$) to indicate equivalence of results. For instance an upward arrow ($\uparrow$) in the RL2 column indicates that RL2 improves on RL1, and the first and second arrows in the RL3 column compare RL3 to RL1 and

Table 1. nDCG values when assessing over all subtopics; arrows indicate improvement(↑), decline (↓) or identical (↔) against previous results lists, first arrow in a cell relates to RL1, second arrow to RL2 and so on. Double arrows (⇑ / ⇓) indicates the comparison is statistically significant returning a two tailed t-test *value* < 0.05.

System	RL1.nDCG	RL2.nDCG	RL3.nDCG	RL4.nDCG
max	0.3433	0.3353	0.3993	0.4118
median	0.2804	0.2918	0.2363	0.2577
min	0.0937	0.0852	0.0000	0.0000
rguBase	0.2669	↓ 0.2594	Not Applicable	Not Applicable
rguPisaSS	0.2669	↓ 0.2528	↓ ↑ 0.2561	↓ ↑ ↑ 0.2577
rguPisaSST	0.2669	↓ 0.2528	↓ ↑ 0.2561	↓ ↑ ↑ 0.2592
	RL1.nDCG@10	RL2.nDCG@10	RL3.nDCG@10	RL4.nDCG@10
max	0.3789	0.4281	0.4307	0.4540
median	0.3232	0.3215	0.3259	0.3407
min	0.1510	0.1432	0.0000	0.0000
rguBase	0.3634	↑ 0.3763	Not Applicable	Not Applicable
rguPisaSS	0.3634	↓ 0.3578	↑ ↑ 0.3735	↑ ↑ ↑ 0.3759
rguPisaSST	0.3634	↓ 0.3578	↑ ↑ 0.3735	↑ ↑ ↑ 0.3773

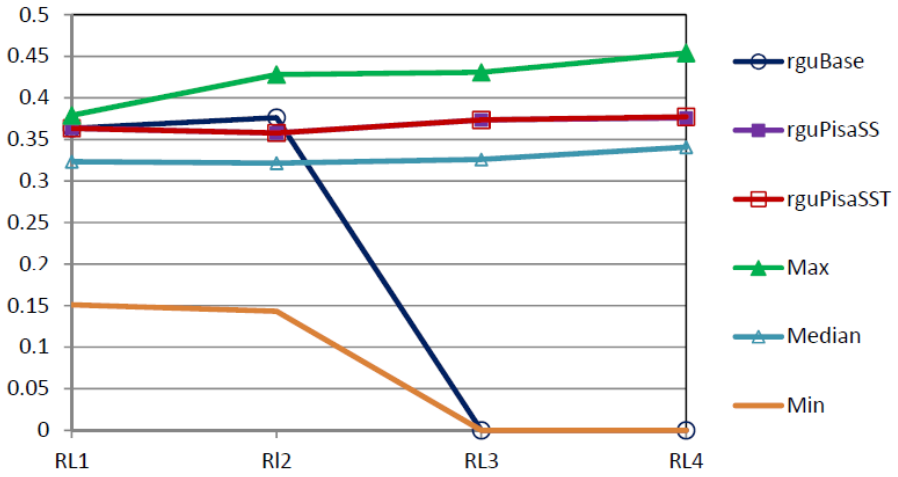

Fig. 4. Graphs showing nDCG@10 for evaluation with all subtopics

RL3 to RL2 respectively. The following two sections outline the results obtained for baseline and SS systems, in cases where we do not explicitly refer to a result as significant it can be assumed that the comparison has returned a t-test value $p > 0.05$.

Table 2. NDCG values when assessing the last subtopic; arrows indicate improvement(↑), decline (↓) or identical (↔) against previous results lists, first arrow in a cell relates to RL1, second arrow to RL2 and so on. Double arrows (⇑ / ⇓) indicates the comparison is statistically significant returning a two tailed t-test $value < 0.05$.

System	RL1.nDCG	RL2.nDCG	RL3.nDCG	RL4.nDCG
max	0.3249	0.3170	0.34611	0.3565
median	0.2562	0.2463	0.2077	0.2169
min	0.0828	0.0737	0.0000	0.0000
rguBase	0.2432	↓ 0.2347	Not Applicable	Not Applicable
rguPisaSS	0.2432	↓ 0.2244	↓ ↓ 0.2223	↓ ↑ ↑ 0.2248
rguPisaSST	0.2432	↓ 0.2244	↓ ↓ 0.2223	↓ ↑ ↑ 0.2260
	RL1.nDCG@10	RL2.nDCG@10	RL3.nDCG@10	RL4.nDCG@10
max	0.2685	0.2954	0.2981	0.2971
median	0.2187	0.1888	0.1859	0.1927
min	0.0781	0.0631	0.0000	0.0000
rguBase	0.2301	↓ 0.2259	Not Applicable	Not Applicable
rguPisaSS	0.2301	↓ 0.2117	↓ ↓ 0.2064	↓ ↓ ↑ 0.2079
rguPisaSST	0.2301	↓ 0.2117	↓ ↓ 0.2064	↓ ↓ ↑ 0.2109

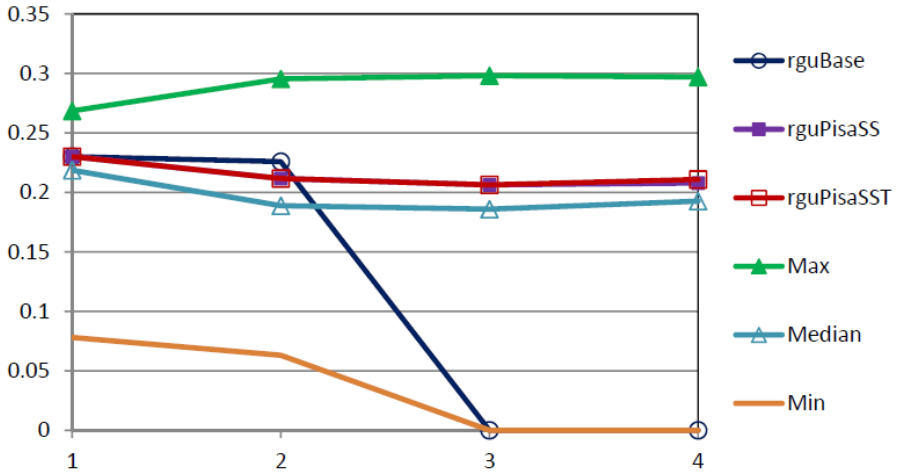

Fig. 5. Graphs showing nDCG@10 for evaluation with last query subtopics

6.1 Expansion with Wikipedia (rguBase)

The retrieval performance of RL2 is worse than RL1 for both evaluation with the last subtopic (Table 2) and nDCG with all subtopics but improves slightly for nDCG@10 (see Table 1). However, these results are better than those from the SS systems when a comparison is made across the column in both tables. This

shows that a lot of relevant documents were from the Wikipedia website and thus reinforces our previous results from last year's session track. Our baseline results are above the median results from all participants (see Figures 4 and 5) despite the simplicity of this technique. The values of RL3 and RL4 are not applicable for this system because we did not submit any results for these criteria using Wikipedia.

6.2 Expansion with Search Shortcuts (rguPisaSS & rguPisaSST)

There is no significant difference when assessing with all subtopics or last query subtopics for both nDCG and nDCG@10 across RL1, RL2, RL3 and RL4. However considering all subtopics, RL4 was generally better than RL2 and RL3 but worse than RL1. Evaluation with nDCG@10 gave us better performance than nDCG for all subtopics as further interactive data are added (RL1 $\rightarrow$ RL4). The only results that met our expectation, albeit slightly, was nDCG@10 for rguPisaSS and rguPisaSST using all subtopics. In this case, there were improvement in performance as more interactive data was incorporated into the systems.

The SS systems were also above the median when compared to other systems as shown in Figures 4 and 5 when evaluated with nDCG@10. We only had a slight improvement from rguPisaSS to rguPisaSST at RL4 for both evaluations with the last and all subtopics. This indicates that incorporating the time did not really make a big difference, at least for our system.

7 Conclusion

This paper provided an overview of the experiments we carried out at the TREC 2011 Session Track. We proposed three different approaches to deal with the tasks introduced this year. While the first one is a pure baseline based on expanding the current query with the term "wikipedia", the last two approaches rely on expanding the current query with the recommendations produced by Search Shortcuts [2], an effective query suggestion technique. The two runs with search shortcuts use different weighting schemes for expansion terms in different ways and depending on the type of available information (document snippets, dwell time, clicks, etc). We intend to improve the last two approaches further by optimising the weight of the expansion terms recommended by the Search shortcuts system.

Acknowledgements

This research is part of the AutoAdapt research project. AutoAdapt is funded by EPSRC grants EP/F035357/1 and EP/F035705/1.

References

1. Baraglia, R., Cacheda, F., Carneiro, V., Fernandez, D., Formoso, V., Perego, R., Silvestri, F.: Search shortcuts: a new approach to the recommendation of queries. In: Proceedings of the third ACM conference on Recommender systems (RecSys'09). pp. 77–84. ACM, New York, NY, USA (2009)
2. Broccolo, D., Marcon, L., Nardini, F.M., Perego, R., Silvestri, F.: Generating suggestions for queries in the long tail with an inverted index. Information Processing and Management (IPM) -, – (2011)
3. Chapelle, O., Metlzer, D., Zhang, Y., Grinspan, P.: Expected reciprocal rank for graded relevance. In: CIKM '09. pp. 621–630. ACM, New York, NY, USA (2009)
4. Cormack, G.V., Smucker, M.D., Clarke, C.L.: Efficient and effective spam filtering and re-ranking for large web datasets. Information Retrieval 14(5), 441–465 (2011)
5. Jones, R., Klinkner, K.L.: Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In: Proceeding of the 17th ACM conference on Information and knowledge management (CIKM'08). pp. 699–708. ACM (2008)
6. Kanoulas, E., Carterette, B., Clough, P., Sanderson, M.: Session track overview. In: Proceedings of Nineteenth Text Retrieval Conference (TREC 2010) (2011)
7. Lavrenko, V., Croft, W.B.: Relevance based language models. In: SIGIR'01. pp. 120–127. ACM, New York, USA (2001)
8. Lucchese, C., Orlando, S., Perego, R., Silvestri, F., Tolomei, G.: Identifying task-based sessions in search engine query logs. In: Proceedings of Third ACM International Conference on Web Search and Data Mining (WSDM'11). pp. 277–286. ACM, New York, NY, USA (2011)
9. Lungely, D., Albakour, M.D., Kruschwitz, U.: The use of domain modeling to improve performance over a query session. In: Proceedings of the ECIR'11 workshop on Information Retrieval Over Query Sessions, SIR'11 (2011)
10. Metzler, D., Croft, B.W.: Combining the language model and inference network approaches to retrieval. Information Processing and Management 40, 735–750 (2004)
11. Radlinski, F., Joachims, T.: Query chains: learning to rank from implicit feedback. In: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. pp. 239–248. KDD '05, ACM, New York, NY, USA (2005), http://doi.acm.org/10.1145/1081870.1081899
12. Robertson, S., Zaragoza, H.: The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends in Information Retrieval 3(4), 333–389 (2009)
13. Robertson, S.E., Kanoulas, E., Yilmaz, E.: Extending average precision to graded relevance judgments. In: SIGIR'10. pp. 603–610. ACM, New York, NY, USA (2010)
14. Yilmaz, E., Kanoulas, E., Aslam, J.A.: A simple and efficient sampling method for estimating ap and ndcg. In: SIGIR '08. pp. 603–610. ACM (2008)