sider this query to have an impression but we do make use of it in our query reformulation pairs unless stated otherwise.

We used the Natural Language Toolkit (NLTK)[3] to remove punctuation and tokenize all textual content, and then stemmed terms using the Porter Stemmer (Porter, 1997). We opted to remove stop words, but bore in mind that this did render some query reformulations as identical to the previous query even if they originally weren't. For instance, in session 95 of the 2012 dataset, q_1 =‘`connecticut fire academy`’ and q_2 =‘`what is the connecticut fire academy`’, yet after stop word removal $q_1 = q_2$. In this case, the reformulation is a more focused query than its predecessor but it nonetheless addresses the same information need with the same core terms. We used the Beautiful Soup HTML Parser[4] to extract textual content from the ClueWeb HTML documents.

We treat each term source (such as a query or snippet) as a bag of words (BoW), even though using n-grams could make our methodology more discernible. For example, in session 285 of the 2014 dataset, q_1 =‘`depression`’ and q_2 =‘`help someone with depression`’. With BoW, we treat the terms ‘`help`’ and ‘`someone`’ separately, and we indeed find examples of the term ‘`help`’ in the snippets for q_1, although erroneously in the context of the webpage (‘`...Help FAQ Advertising...`’ at rank 3) rather than that implied by the query. Here, a bigram would distinguish ‘`help someone`’ in the correct context. Nonetheless, all of the similarity measures we use operate on a BoW model, and given that we typically only see 1 or 2 terms being added or removed from adjacent queries in a session, a unigram model is sufficient in this case.

Our methodology concerns the analysis of text similarities. We measure the similarities of queries using the following formulae:

$$Jaccard(Q_1, Q_2) = \frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|} \tag{1}$$

$$Cosine(\overrightarrow{q_1}, \overrightarrow{q_2}) = \frac{\overrightarrow{q_1} \cdot \overrightarrow{q_2}}{\|\overrightarrow{q_1}\| \cdot \|\overrightarrow{q_2}\|} \tag{2}$$

where q_1 and q_2 are queries (or any other term source). *Jaccard* similarity is commonly used in measuring set similarity, in this case sets of terms, and *Cosine* similarity is widely used in the vector space model in IR.

4 Term Retention and Removal

In our first analysis we investigate the term actions *retention* and *removal*. These two actions are only applied to terms found in the user's query t_n, where *retention* means that $t_n \in Q_{n+1}$ and *removal* is when $t_n \notin Q_{n+1}$.

[3] http://www.nltk.org/

[4] http://www.crummy.com/software/BeautifulSoup/

Table 3 Average number of terms retained, removed or added from $q_n \rightarrow q_{n+1}$ and the similarity between the two queries across TREC Session Track datasets.

	TREC Session Track				
	2011	2012	2013	2014	Combined
$Jaccard(\overrightarrow{q_n}, \overrightarrow{q_{n+1}})$	0.49	0.52	0.50	0.51	0.50
$Cosine(Q_n, Q_{n+1})$	0.60	0.65	0.62	0.63	0.63
# terms *retained* from $q_n \rightarrow q_{n+1}$	2.12	2.29	2.28	2.10	2.13
# terms *removed* from $q_n \rightarrow q_{n+1}$	1.20	1.05	1.20	1.11	1.12
# terms *added* from $q_n \rightarrow q_{n+1}$	1.33	1.35	1.33	1.21	1.24

We measured the average number of terms retained, removed or added and the average Jaccard and Cosine similarity between adjacent queries found in sessions in the TREC datasets, our results are in Table 3. We see that adjacent queries are similar to one another, with high similarity scores and term retention. We note that measures are generally consistent across the individual datasets and their combination, and so the remainder of our analyses will be conducted on the combined dataset. We find that across all datasets, an average of 63% of the terms in q_{n+1} can be found in q_n, where 66% of its terms are *retained* (2.13 terms), 34% of terms are *removed* (1.12 terms) and 1.24 terms are added. 33% of the time the reformulation contains all of the terms found in the original query. Retained terms clearly make up a large proportion of a reformulation and are indicative of the core terms defining the user's information need.

An important observation is that on average the length of queries increases from 3.25 terms to 3.37 terms, meaning that it cannot always be possible to source q_{n+1} terms from q_n. To determine if this relationship holds throughout a session, we found the average query length at each impression position for a number of different session lengths (see Figure 1). Our results show that for shorter sessions (2 - 4 impressions) query size does appear to marginally increase, for medium session lengths (5 - 7 impressions) the query size initially increases to a point and can start to decrease, and for longer sessions (8 - 10 impressions) the query length varies unpredictably, presumably due to the small population sizes. Medium and longer sessions are likely to contain shifts in information need (for example, between queries 4 and 5 in Table 1), which may explain the variability of query length with increased impression position. It is clear from these results that reformulations can gain or lose terms depending on its position in a session.

In Figure 2 we measured the similarity between query reformulations and their preceding query at each impression position. In our previous analysis we found that impression position affected query length (subject to session length), so here we investigate if this also holds for query similarity. The main conclusion we can draw is that the results are too variable to discern a pattern, with no clear trend for increasing or decreasing similarity. What this tells us is that throughout a session, queries are generally similar to their reformulations regardless of position in the session.

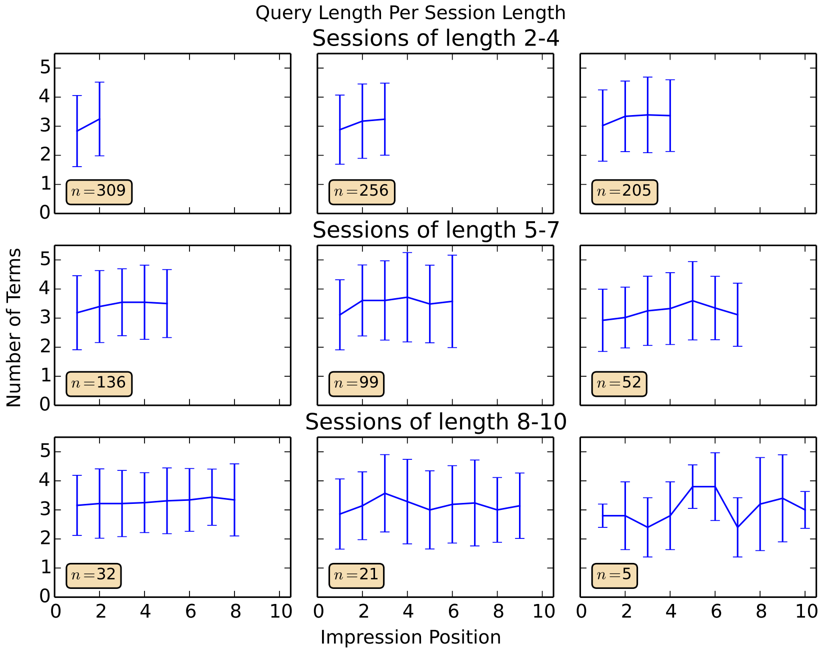

Fig. 1 Plots of the average number of terms in queries at different impression positions in a session, for different lengths of session. The number of instances of each session length are labeled as n in each subplot.

Nonetheless, we do expect information needs to change throughout a session and when that happens the similarity between adjacent queries should change. For instance, in Table 1 the average similarity scores between all adjacent queries are $Jaccard = 0.44$ and $Cosine = 0.57$, but between queries 4 and 5, the shift in query intent is captured in the change in similarity scores, calculated as $Jaccard = 0.17$ and $Cosine = 0.29$, a noticeable departure from the average.

In Figure 3 we show that core query terms do not remain constant throughout a session, indicating that the terms used in queries are always progressively changing. In this instance we picked cosine similarity although we observe the same trend for Jaccard similarity. We see that queries occurring on either side of the 'fixed' query q_x are the most similar but queries further away in the session become more dissimilar. This behavior holds regardless of the position of q_x in the session. This and the previous result demonstrates one of the key motivations of our methodology, that there does not exist a set of 'core' terms that represent the user's information need throughout the session, instead, the query and its core terms evolve as the user's information need changes. Queries at the start of a session can be very different from those at the end, and as such, term retention and removal are useful locally with adjacent queries but less so across the whole session.