Exploiting Extended Search Sessions for Recommending Search Experiences in the Social Web

Zurina Saaya, Markus Schaal, Maurice Coyle, Peter Briggs and Barry Smyth

CLARITY: Centre for Sensor Web Technologies
School of Computer Science and Informatics
University College Dublin, Ireland
`firstname.lastname@ucd.ie`
`http://www.clarity-centre.org`

Abstract. HeyStaks is a case-based social search system that allows users to create and share case bases of search experiences (called *staks*) and uses these staks as the basis for result recommendations at search time. These recommendations are added to conventional results from Google and Bing so that searchers can benefit from more focused results from people they trust on topics that matter to them. An important point of friction in HeyStaks is the need for searchers to select their search context (that is, their active stak) at search time. In this paper we extend previous work that attempts to eliminate this friction by automatically recommending an active stak based on the searchers context (query terms, Google results, etc.) and demonstrate significant improvements in stak recommendation accuracy.

Keywords: social search, community recommendation

1 Introduction

Over the past few years there has been a growing interest in the application of case-based reasoning techniques to the type of *experiences* and *opinions* that are routinely captured as web content. The modern web is characterized by a proliferation of user-generated content. On the one hand we are all familiar with user generated content in the form of blog posts, online reviews, comments and ratings. On the other hand there is an equally rich tapestry of *implicit* experiential signals created by the actions of web users: the links people follow as their navigate, the results we select when we search, the pages we bookmark and share, and the movies and music we play. Collectively this content, and our actions as we consume and share it, encode our experiences and these experiences constitute the raw material for reuse as evidenced by recent work in the area of *WebCBR* and the *Experience Web* [16], [22].

We are also interested in the experience web, specifically in the search experiences of users and the opportunity to reuse these experiences in order to improve the effectiveness of mainstream web search. Modern search engines continue to struggle when it comes to delivering the right results to the right users at the right time. This is particularly acute in today's culture of sophisticated search engine optimization (SEO) techniques and so-called *content farming* strategies, which are designed to boost the rank of targeted results, often to the detriment of the individual searcher. Much has been written about the need for a more *personalized approach to web search*, see [5], [7], [17]. One particular approach to improving web search that has been gaining traction recently is evidenced by recent moves by mainstream search engines to introduce an element of so-called *social search* into their workflow, by incorporating results that have originated from the searcher's social network (e.g. Twitter, FaceBook, Google+), borrowing ideas from work on *collaborative web search*, see [20], [15], [2].

In this work we will focus on one particular approach to collaborative web search as presented previously in work by [21] and implemented in a system called HeyStaks. HeyStaks integrates collaborative web search into Google and Bing via a browser plugin. HeyStaks is further informed by the recent interest in curation on the web as evidenced by the emergence of content curation services such as Pinterest, Clipboard, ScoopIt etc; see [14]. Briefly, HeyStaks allows users to curate and share collections of search experiences called *staks*; each stak is essentially a case base of search experiences on a given topic. As a stak member searches, in the context of a given stak, any results they select are added to the stak. Then, in the future, when other members search for similar queries they may be recommended these results, in addition to the standard Google or Bing results. For example, consider a small group of college students planning a vacation together. One of the students might create a *travel* stak and share it with the others. Over time their vacation-related searches will add valuable queries and results to this stak and other members will benefit from these experiences by getting recommendations from this stak during their searches. We will review the operation of HeyStaks in more detail in Section 3.

We will focus on a specific problem faced by HeyStaks users, namely the selection of an appropriate context (stak) for their target search. Currently, HeyStaks users need to select one of their search staks at search time to ensure that any queries and selections are stored in the correct context. If users forget to select a stak, which they frequently do, then search experiences can be mis-recorded, compromising stak quality and leading to poor recommendations in the future. Recently [19] proposed an alternative to manual stak selection by using textual CBR methods to recommend staks at search time. While results were promising they were not at the level necessary to use in practice. For example, to be practical it is necessary to be able to recommend an appropriate stak 80-90% of the time. However, the work of [19] achieved recommendation success rates of less than 60%. This initial solution based its recommendations on a straightforward term-overlap between the terms associated with the searcher's current query (e.g. query terms and/or terms from the results retrieved from the underlying mainstream search engine) and terms that reflect stak topics (e.g., terms associated with queries and or pages that make up the stak). The main contribution of this work is a two-part extension of the work of [19]. First we look at the potential to profile a searcher's query across multiple query instances (a so-called *search session*), since searchers often submit a series of related queries (and receive different result lists) before they find what they are looking for. Second, we describe a technique for weighting the relative importance of search session terms during recommendation. We go on to present a set of results based on live-user usage of HeyStaks to demonstrate the potential of these new techniques relative to the benchmark described by [19].

2 Related Work

There is a history of using case-based methods in information retrieval and web search. For example, the work of [18] looks at the application of CBR to legal information retrieval (see also [3]), and [6] describe a case-based approach to question-answering tasks. Similarly, in recent years there has been considerable research looking at how CBR techniques can deal with less structured textual cases. This has led to a range of so-called *textual CBR* techniques [13][23].

In the context of Web search, one particularly relevant piece of work concerns the *Broadway* recommender system [12] and a novel query refinement technique that uses case-based techniques to reuse past query refinements in order to recommend new refinements. Broadway's cases reference a precise experience within a search session and include a problem description (made up of a sequence of behavioural elements including a sequence of recent queries), a solution (a new query refinement configuration), and an evaluation (based on historical explicit user satisfaction ratings when this case was previously recommended). The work of [8] apply CBR techniques to Web search in a different way. Very briefly, their *PersonalSearcher* agent combines user profiling and textual case-based reasoning to dynamically filter Web documents according to a user's learned preferences.

More recently researchers have applied CBR concepts to web search. For example, [4] introduced *collaborative web search* (CWS) and the idea of reusing the search experiences of communities of like-minded searchers as implemented in the form of a system called *I-SPY*. In short, each community is associated with a case base of past *search cases*, with each case taking the form of a set of query terms (the problem specification) and a selected result (the problem solution). When presented with a new target query the I-SPY system retrieves a set of cases with similar queries and rank orders a set of corresponding results, recommending the top ranking results which have been frequently selected for similar queries by the community during past searches. This approach was recently expanded on by the work of [21] in the HeyStaks system, which allowed users to create and share their own case bases of search experiences. HeyStaks also extended search case representations to include snippet information, tags, and sharing and voting signals, in addition to simple query terms, in order to facilitate a more flexible approach to case similarity and retrieval. We will review HeyStaks in the following sections as it forms the basis of the work presented

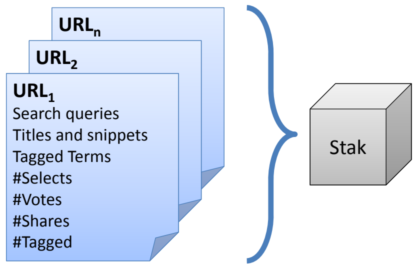


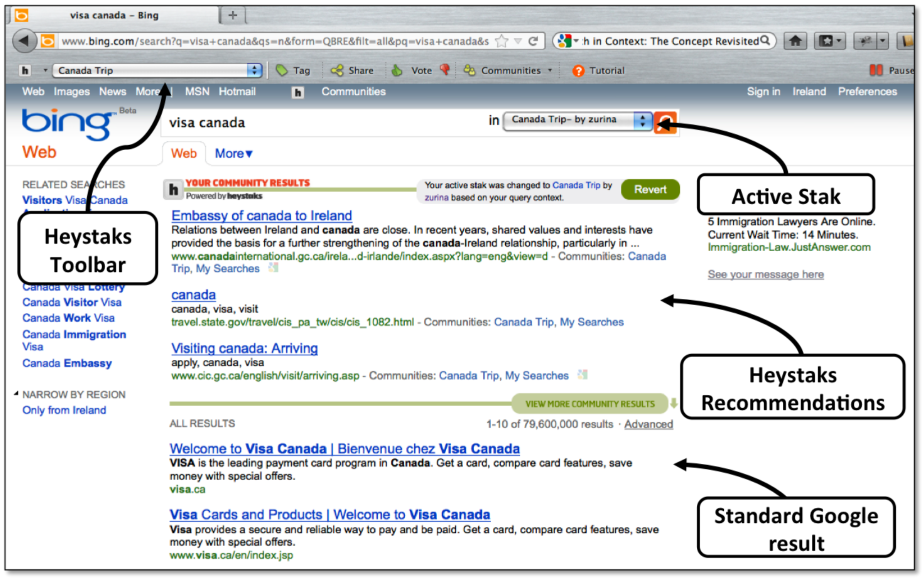
Fig. 1. Search Experiences in HeyStaks

in this paper and then proceed to describe the stak recommendation task and evaluate our extended solution.

3 HeyStaks: A Case-Based Approach to Social Search

HeyStaks combines a number of ideas to deliver an improved search experience for users. First and foremost it is based around the notion of *collaboration*, namely that it is useful for users to be able to collaborate as they search. Second it emphasizes the importance of *curation* and the willingness of interested users to create and maintain collections of topical content. Thirdly, it stresses the importance of *integration* by delivering social search within the context of an existing search service such as Google, by integrating with Google via a browser plugin. Bringing all of these ideas together HeyStaks allows communities of like-minded users to create and share curated repositories of search experiences, which deliver targeted recommendations to community members as they search, in addition to the organic results of Google, Bing or Yahoo.

The central idea in HeyStaks is the notion of a *search stak*. A stak is a named collection of search experiences. It is represented as a case base of search cases. Users can create and share their own search staks or they can join those created by others. Staks will typically be created around a topic that matters to a group of users, perhaps an upcoming vacation. At their core staks contain URLs for web pages that have been found during search sessions. Each URL is essentially the solution of a search case and is associated with a set of *specification* features that capture the different ways in which this case has been located in the past. For example, these features will typically include the terms of any queries that led to this URL being selected. Similarly, any snippet terms associated with the URL by some underlying search engine can also be used as part of the case specification. HeyStaks users can tag, rate, and share URLs too and this information will also be captured as part of a given URL's specification; see Figure 1. In this way each URL is associated with a rich set of *search experiences* that have led to its selection and these features can be used during future searches as a means to decide whether or not the URL should be recommended to the future searcher.


visa canada – Bing
www.bing.com/search?q=visa+canada&qs=n&form=QBRE&filt=all&pq=visa+canada&s
h in Context: The Concept Revisited
Canada Trip
Tag
Share
Vote
Communities
Tutorial
Pause
Web Images News More MSN Hotmail Communities
Sign in Ireland Preferences
bing Beta
Web
visa canada
in Canada Trip– by zurina
Web More
Heystaks Toolbar
Active Stak
Heystaks Recommendations
Standard Google result
Revert
VIEW MORE COMMUNITY RESULTS

RELATED SEARCHES
Visitors Visa Canada
Canada Visa Lottery
Canada Visitor Visa
Canada Work Visa
Canada Immigration Visa
Canada Embassy

NARROW BY REGION
Only from Ireland

YOUR COMMUNITY RESULTS
Powered by heystaks

Your active stak was changed to Canada Trip by zurina based on your query context.

Embassy of canada to Ireland
Relations between Ireland and **canada** are close. In recent years, shared values and interests have provided the basis for a further strengthening of the **canada**-Ireland relationship, particularly in ...
www.**canada**international.gc.ca/irela...d-irlande/index.aspx?lang=eng&view=d - Communities: Canada Trip, My Searches

canada
canada, visa, visit
travel.state.gov/travel/cis_pa_tw/cis/cis_1082.html - Communities: Canada Trip, My Searches

Visiting canada: Arriving
apply, canada, visa
www.cic.gc.ca/english/visit/arriving.asp - Communities: Canada Trip, My Searches

ALL RESULTS
1-10 of 79,600,000 results · Advanced

Welcome to **Visa Canada** | Bienvenue chez **Visa Canada**
VISA is the leading payment card program in **Canada**. Get a card, compare card features, save money with special offers.
visa.ca

Visa Cards and Products | Welcome to **Visa Canada**
Visa provides a secure and reliable way to pay and be paid. Get a card, compare card features, save money with special offers.
www.**visa**.ca/en/index.jsp

5 Immigration Lawyers Are Online.
Current Wait Time: 14 Minutes.
Immigration-Law.JustAnswer.com
See your message here

Fig. 2. The searcher is looking for visa information to enter Canada but Bing responds with results related VISA payment system. HeyStaks recognises the query as relevant to the searcher's *Canada Trip* stak and presents a set of more relevant results drawn from this stak.

The full details of staks and their search cases have been covered elsewhere (see [21]) and are beyond the scope of this paper.

Fig. 2 shows HeyStaks in operation. It shows a searcher, looking for Canadian visa information, benefiting from recommendations made by a group of friends who share a search stak called *Canada Trip*. The HeyStaks browser toolbar is shown, which provides access to key functionality such as tagging, rating, sharing etc. In this example, the user has selected the *Canada Trip* and a set of recommendations are shown alongside the standard Bing result list. These recommendations have been inserted into the standard Bing results page via the HeyStaks toolbar and the recommendations have been selected based on the past search experiences of other stak members. In this way our searcher benefits from the wisdom of people they trust on a topic that matters to them while they search. Once again the full details of this recommendation process are beyond the scope of this paper but they have been discussed in detail in [21]

One notable feature of this example is that the screenshot shows the *Canada Trip* stak being suggested to the user at search time; notice the message *"Your active stak was changed to Canada Trip"*. This is important for this paper because we are focused on automating this type of recommendation to the user, as opposed to recommending search results as per [21] currently these stak recommendations are very limited in HeyStaks. It is important for the user to be using the right stak for a given search if they are to benefit fully from HeyStaks' recommendations. In the past this has been a largely manual process, meaning the user was expected to select the stak at search time, which is far from ideal since it introduces an additional point of friction into the overall proposition.

4 Recommending Search Case Bases

The recommendation of prior search results depends crucially on the recommendation of a suitable stak to the searcher at search time. The importance of selecting the correct stak is twofold. On the one hand, the detection of staks related to the current query provides the basis for result recommendations for the user and if this detection does not work properly then in all likelihood any recommendations made by HeyStaks to the user will be unlikely to be effective. But much more importantly, the current active stak provides a context for any search experiences such as selections, tagging, rating that the users make during the search session, etc. If the wrong stak is chosen, search experiences will be stored in an inappropriate stak, thus polluting this incorrect stak with irrelevant results, and preventing the case-base to gain maturity through learning based on the user's current session.

How then can we select and recommend a stak at search time? Prior work of [19] has looked at how to profile a stak and, given a current search query or partial search experience, how to identify those staks that are most likely to correspond to the current search needs. A summary of this stak recommendation process is presented in Figure 3. Briefly, for each user a new case base called the *stak summaries case base* (SSCB) is produced. In this case base each *stak case* corresponds to a single stak. In other words it is a case that is produced from a combination of the individual *search cases* in the corresponding stak case base. In effect, the specification part of each stak case is the combination of the specifications of the corresponding stak's search cases and the solution of the stak case is the corresponding stak id. In this way, stak cases are associated with the queries, snippets, URLs etc that are the basis of the search experiences within individual staks.

At search time we can use information about the searcher's current search context in order to retrieve a set of similar stak cases from the SSCB, or preferably automatically switch the user into a stak based on the most similar stak case. What information can be used to do this? In [19] a number of sources of information were considered including the searcher's current query, the snippet text for any results returned by the underlying search engine, the URLs of these results, and popularity information for the user's staks. This information can be used to build a *stak query* S_Q for the user and then we can score each stak case in the SSCB using a scoring metric such as that shown in Equation 1 to produce a ranked list of stak recommendations as per Equation 2.

$$Score(S_Q, S_C, SSCB) = \sum_{t \epsilon S_Q} tf(t, S_C) \times idf(t, SSCB) \tag{1}$$

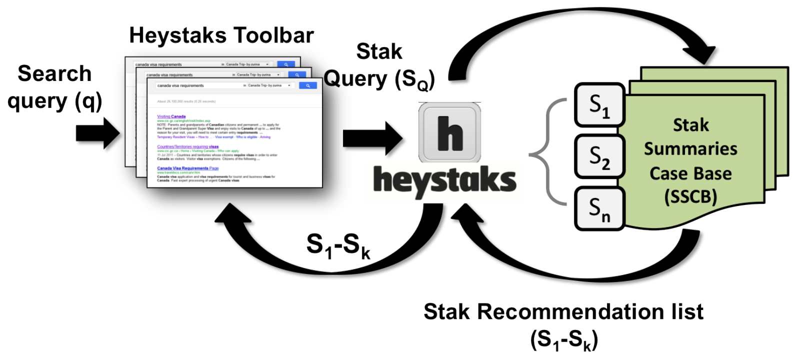


Fig. 3. An overview of the stak recommendation process

$$RecList(S_Q, S_U, SSCB) = \begin{array}{l} SortDesc(Score(S_Q, S_C, SSCB)) \\ \forall S_C \epsilon S_U \end{array} \tag{2}$$

Thus the user's stak query S_Q is made up of a set of terms (that may include query and snippet terms, URLs etc) and we can use TF-IDF [9] to calculate the relevance of a candidate stak S_C (from those staks the user is a member of, S_U) to S_Q; see Equation 1, which gives a higher weighting to S_Q terms that are common in S_C but uncommon across SSCB as a whole. By using different combinations of features for the stak query S_Q we can implement and test a number of different stak recommendation strategies as discussed by [19]. In what follows in this work we will focus on the combinations shown in Table 1.

Strategy	Description
URL	URLs from the result-list
Snippet	Page titles and snippets from the result-list
Query	User's search query
URLSnippetQuery	Combination of URLs, search query, page titles and snippets
Popular	Most frequent stak for the user
URLSnippetQueryPopular	Combination of all strategies

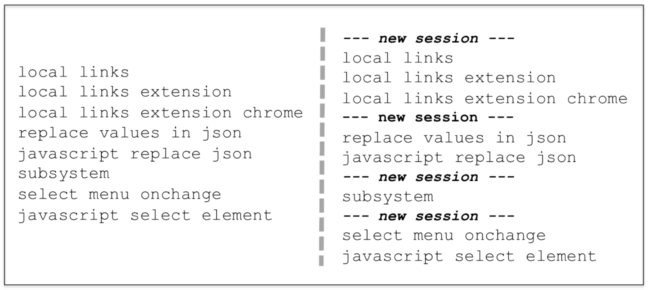
Table 1. Strategies for Staks Recommendation

5 Extending Stak Recommendation

The approach of [19] is limited in a number of important respects. Firstly each search is considered to be an atomic (singleton) session, which is not the way that people search in practice. Very often a searcher will require a few (related) queries to find what they are looking for (see [1]) and this means that we are not limited to using a single query (and its attendant data) for stak recommendation. In principle it is possible to assemble a stak query from the information contained in extended search sessions that span multiple queries, and in so doing provide a richer S_Q as the basis for recommendation. Moreover, the work of [19] did not consider the weighting of S_Q features/terms. But if we are constructing stak queries across multiple sessions then frequently recurring terms can be considered more important within S_Q, for example, and this information can be used to further enhance stak recommendation. We will develop and evaluate both of these ideas in the remainder of this paper.

5.1 Harnessing Extended Sessions

Given that many search sessions will span multiple queries it is natural to consider the possibility of using additional information from an evolving search session as the basis for stak recommendation. For example, while it might not be possible to reliably recommend the correct stak on the first query, the availability of an additional query (and its associated URLs and snippets) may improve recommendation quality. Thus, an alternative stak recommendation approach should build each stak query S_Q by aggregating the information that is available across related searches. Of course to do this it is necessary to be able to identify an extended search session. For the purpose of this work we use the method introduced by [11] (see Fig. 4), which effectively groups queries from a search log based on a simple term-overlap threshold.



```
local links
local links extension
local links extension chrome
replace values in json
javascript replace json
subsystem
select menu onchange
javascript select element
```

```
--- new session ---
local links
local links extension
local links extension chrome
--- new session ---
replace values in json
javascript replace json
--- new session ---
subsystem
--- new session ---
select menu onchange
javascript select element
```

Fig. 4. (Left) A sequence of queries submitted by one user. (Right) Sessions obtained based on shared query terms as per [11].

Thus, the stak query, as defined in the previous section, can now be adapted to cover sessions with multiple queries as $S^*_Q = \{S^1_Q, ..S^n_Q\}$. And in turn we can apply the stak recommendation techniques to these extended stak queries to investigate their impact on overall recommendation accuracy. In this way, as a search session evolves, the stak recommendation system has access to an increasing volume of relevant information as the basis for recommendation. The intuition is that this will improve recommendation quality, which we will come to test in due course.

5.2 Session-based Term Weighting

The second limitation of the work presented in [19] is that the elements of a stak query are all assumed to be equally important. In other words there is no relative weighting of stak query terms even though, as we accumulate information across extended search sessions, there is an opportunity to introduce a weighting model into the stak recommendation process. Simply put, we can use term frequency information (calculated across a search session) as term weights, $W(t, S_Q^*)$, during recommendation. In this way the terms that frequently recur in the searchers queries (or in the snippet texts of the organic search results) are deemed to play a more important role during stak selection according to the new scoring function presented in Equation 3.

$$Score(S_Q^*, S_C, SSCB) = \sum_{t \epsilon S_Q^*} tf(t, S_C) \times idf(t, SSCB) \times W(t, S_Q^*) \tag{3}$$

Once again the intuition is that the combination of extended search sessions as a richer source of query information combined with the availability of term weights should help to improve overall recommendation effectiveness. The next section tests this hypothesis in detail by comparing a variety of different recommendation strategies, using different sources of query information, and combining extended sessions and term weighting.

6 Evaluation

In the previous work of [19] a stak recommendation strategy was tested using only the information from singleton sessions. In what follows we will adopt a similar methodology but look at the effectiveness of recommendation when using the extended session and term-weighting techniques described above.

6.1 Dataset and Approach

The dataset comes from the HeyStaks anonymous usage logs based on a group of 28 active users, who are members of approximately 20 staks each, and who have each submitted at least 100 queries. For the purpose of this evaluation we limit our interest to only those sessions that are associated with at least one non-default search stak. This is important because it means that we can focus on search scenarios where the user has actively selected a specific stak during their search session. This selected stak is used as the ground-truth against which to judge our recommendation techniques; in other words, we are using information from the user's search session to make a stak recommendation and

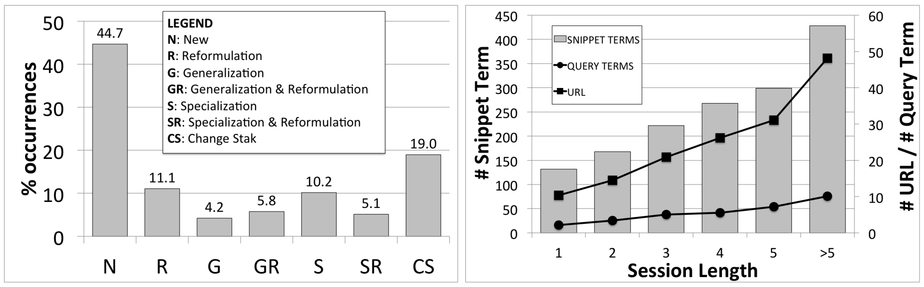


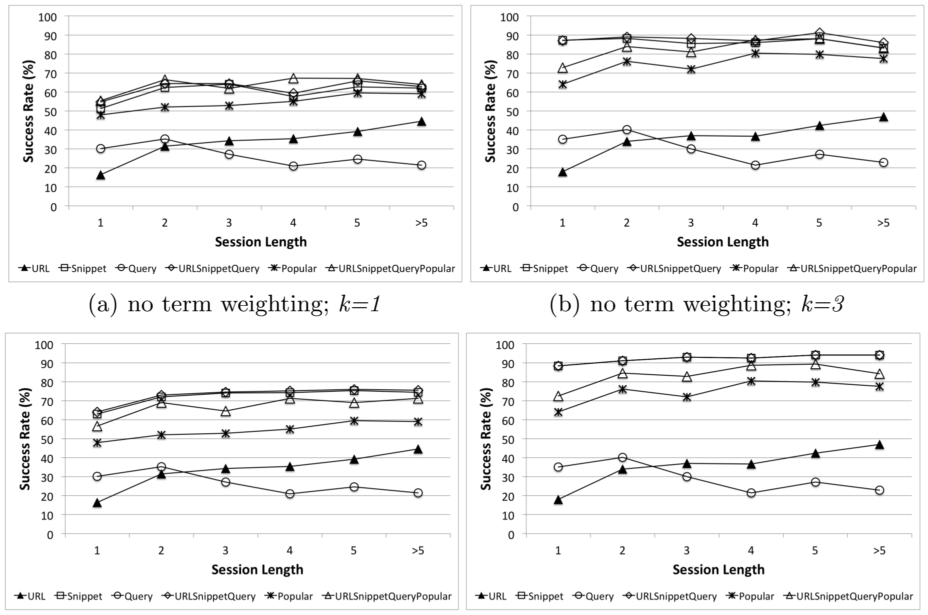
Fig. 5. Summary of session data; (a) query modification classes and (b) average unique URLs, query terms and snippet terms across different session length

we compare this to the actual stak that they chose at search time. If their chosen stak matches the recommendation then the recommendation is deemed to be correct or successful. Arguably, this is quite a strict measure of success. After all, users sometimes join a number of related staks and even if the correct stak is not recommended a related one might be, which would probably still be useful to the searcher. However, we ignore these *near-miss* recommendations and treat them as failures for the purpose of this strict evaluation. According to the above criteria our test dataset covers 10,177 individual searches which have been grouped into 4,545 sessions, and the average each user has submitted 364 queries in 162 sessions.

6.2 Session Data

Before describing the results of the session-based evaluation it is useful to look at the relationship between consecutive queries, q_i and q_{i+1}, from the same user and session, to get a sense of the type of modifications and refinements that searchers tend to make within extended sessions. The following modification classes are based on those presented in [10]:

- *New* – initial query in a session.
- *Reformulation* – the current query is on the same topic as the previous query and both queries contain common terms. (add some terms and removed others from the previous query and both queries still have some common terms)
- *Generalization* – the current query is in the same topic as previous query, but the searcher seeking more general information (remove terms from previous query)
- *Specialization* – the current query is on the same topic as the previous query, but the search is now seeking more specific information. (adding new terms to the query)
- *Change Stak* –the current query identical with the previous query, but the stak has been changed



(a) no term weighting; *k=1* (b) no term weighting; *k=3*

(c) with term weighting; *k=1* (d) with term weighting; *k=3*

Fig. 6. Success Rate for both approaches when $k = 1$ and $k = 3$

Fig. 5(a) shows the frequency of these query modification classes. We see that users frequently change their staks during a session (19% of the time) and that about 36% of the modifications involve changes to the terms in the query, which will ultimately lead to changes in the result-list returned to users, and so provides a strong indication that leveraging these extended sessions will deliver a richer source of information for stak recommendation. In Fig. 5(b) we present the unique number of query and snippet terms, and URLs, for different session lengths and we can see that there is a steady increase in the quantity of this information as session length grows. However, it is worth highlighting that, for example, doubling the session length, does not deliver twice the number of unique query or snippet terms or unique URLs; the reason being, of course, that minor modifications to the query will not radically change the new result-list.

6.3 Experimental Setup

We are primarily concerned with the accuracy of our three basic recommendation strategies *Query, Snippet, URL*, which differ based on the type of basic information used for a stak query. As per [19] we also consider the combination of these techniques and further combine them with the baseline stak *popularity* strategy, which recommends the stak most frequently used by the user. In total we look at 6 different strategies; see Table 1 earlier.

To evaluate these alternatives, we generate a recommendation list for each of the 4,545 sessions and compute the percentage of times (*success rate*) that the known active stak (ground-truth) is recommended in the top k stak recommendations (here we look at $k = 1$ and $k = 3$). We calculate this success rate across sessions of different lengths, both with and without term weighting.

6.4 Success Rate vs. Session Length

The results are presented in Fig. 6(a-d). Similar to the findings of [19], techniques such as *URL* and *Query* perform poorly, while *Popularity* and *Snippet*, and combinations thereof, perform well. This is true across all techniques and session lengths. However, there is a wide variety in absolute success rates.

As we can see from the graphs, increased session length generally implies improved success rates, especially when comparing sessions of length 1 (singleton sessions as per [19]) with sessions of length 2. It is particularly important to pay special attention to the $k = 1$ results because the ideal strategy for HeyStaks is to automatically switch the user into a correct stak, rather than present a set of (> 1) stak recommendations for the searcher to choose from. For $k = 1$, we can see for example that the best performing singleton strategies deliver success rates in the region of 51-56%; see, for example, Figure 6(a). By comparison, when we consider sessions of length 2 this tends to increase to 61-66%, a relative improvement of just under 20%. However, this rate of improvement is not sustained over longer search sessions and by and large the success rates for longer sessions are no higher than those for sessions of length 2. In other words, despite the availability of additional query, URL, and snippet data in longer sessions of length 3, 4, 5 etc., this extra data does not seem to help from a stak recommendation viewpoint. Another influencing factor is probably that searchers that *require* long sessions (> 2 queries) are probably fundamentally harder to satisfy than those that require just 2 queries and so success is likely to be more elusive. At the moment the test data contains a mixture of these data and so it is a matter for future work to further consider this explanation.

6.5 Term Weighting

The results of adding term weighting are presented in Figure 6(c, d) and further analysed in Figure 7. One of the best performing strategies when using term weighting, *URLSnippetQuery*, achieves a success rate of more than 70% for session lengths > 2, at $k = 1$); see 6(c) for example. This is an improvement of up to 17% in comparison to the results without term weighting.

Figure 7 looks at the relative performance of the three best performing techniques using term weighting and compares them to their corresponding results without term weight, across different session lengths; we focus on $k = 1$ recommendations only in this instance but comparable results are found for $k = 3$. The results show that the relative improvement due to term weighting falls off sharply with increasing session lengths. For example,*Snippet* achieves a success rate of about 52% for sessions of length 1, rising to about 62% for sessions of

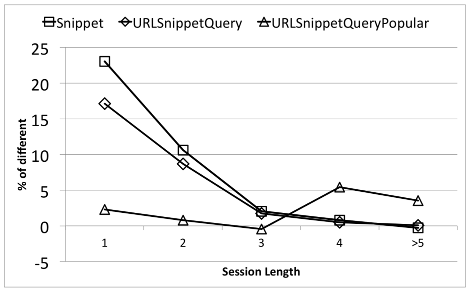
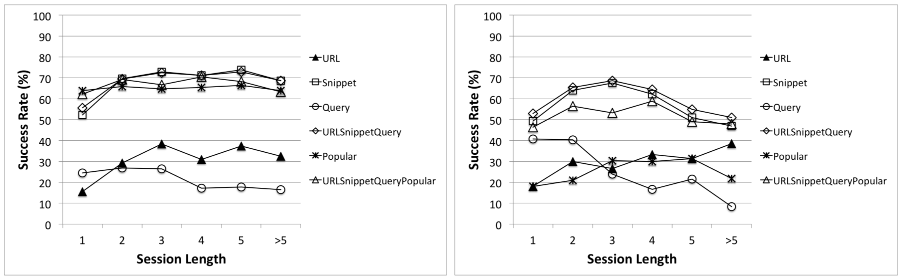


Fig. 7. Difference of success rate between standard session-based and term weighting



(a) Large staks with term weighting, k=1 (b) Small staks with term weighting, k=1

Fig. 8. Large vs. Small Search Staks

length 2, with term weighting is not used; see Figure 6(a). When term weighting is used these success rates are 63% and 72%, respectively, leading to relative improvements of about 23% and 11% as shown in Figure 7. These relative improvements continue to decline for *Snippet* as session lengths increase, as can be seen in Figure 7, and similar relative improvements, and subsequent declines, are also observed for *URLSnippetQuery* and *URLSnippetQueryPopular* as indicated.

These results help to clarity that, in combination, term weighting and extended sessions do have a positive impact on overall success rates. In absolute terms this combination is beneficial but scale of the relative benefit decreases with session length.

6.6 Success Rate of Stak Size

It is interesting to consider the influence of stak size on recommendation success rates. The majority of staks in this study (94%) contain a relatively few pages (1-500 URLs) which we expect to provide a relatively weak recommendation signal. As HeyStaks matures we can expect users to develop more mature staks and so it is appropriate to evaluate the relationship between recommendation success and stak size. To test this, we juxtapose the recommendation success rates by dividing the data according to the stak size into *small* (< 500 URLs) and *large* (>= 500 URLs), for $k = 1$ and with term weighting; see Figure 8.

Clearly there are differences in accuracy between small and large staks. Larger, more mature staks enjoy higher success rates across the various recommendation techniques and session length settings. For example, looking at *URLSnippetQueryPopularity*, we see a success rate of 63-70% for large staks (Figure 8(a)) compared to only 47-60% when used with small staks (Figure 8(b)). This is encouraging because, from a engineering standpoint, these higher success rates suggest it may be practical to implement a reliable automatic stak switching policy, for larger staks which contain more than 500 URLs.

7 Conclusions

In this paper we have reconsidered the stak recommendation challenge faced by HeyStaks, a case-based social search solution. Our main contribution has been to extend the work of [19] by exploring a number of novel stak recommendation strategies that take advantage of the additional information that is available as a source of context across extended search sessions. The results, based on live-user search logs, suggest that recommendation success can be improved by using extended search session data, albeit with certain caveats. For example, the benefit of using extended search sessions is maximized as when we consider the difference between singleton sessions and sessions of length 2. That being said, it is to the advantage of HeyStaks that these benefits are maximized for shorter sessions because these sessions are more frequent than longer sessions. Moreover, the relative improvements that we have found in stak recommendation accuracy, compared to the past work of [19], suggest that this new approach may have practical merit in a deployment setting, certainly for large stak sizes.

Acknowledgments

This work is supported by Science Foundation Ireland under grant 07/CE/I1147, HeyStaks Technologies Ltd, Ministry of Higher Education Malaysia and Universiti Teknikal Malaysia Melaka.

References

1. Amanda Spink, Tom Wilson, D.E., Ford, N.: Modeling Users' Successive Searches in Digital Environments. D-Lib Magazine (1998)
2. Amershi, S., Morris, M.R.: CoSearch: A System for Co-located Collaborative Web Search. In: Proceeding of the 26th Annual SIGCHI Conference on Human Factors in Computing Systems. pp. 1647–1656. CHI '08, ACM, New York, NY, USA (2008)
3. Ashley, K.D.: Modeling Legal Arguments: Reasoning with Cases and Hypotheticals. MIT Press, Cambridge, MA, USA (1991)
4. Balfe, E., Smyth, B.: Case-Based Collaborative Web Search. In: Funk, P., Gonzalez Calero, P.A. (eds.) Advances in Case-Based Reasoning, Lecture Notes in Computer Science, vol. 3155, pp. 1015–1050. Springer Berlin / Heidelberg (2004)

5. Bharat, K.: SearchPad: Explicit Capture of Search Context to Support Web Search. Computer Networks 33(1-6), 493 – 501 (2000)
6. Burke, R., Hammond, K., Kulyukin, V., Tomuro, S.: Question Answering from Frequently Asked Question Files. AI Magazine 18(2), 57–66 (1997)
7. Dou, Z., Song, R., Wen, J.R.: A Large-scale Evaluation and Analysis of Personalized Search Strategies. In: WWW 07: Proceedings of the 16th International Conference on World Wide Web. pp. 581–590. ACM Press, New York, NY, USA (2007)
8. Godoy, D., Amandi, A.: PersonalSearcher: An Intelligent Agent for Searching Web Pages. In: Monard, M.C., Sichman, J.S. (eds.) IBERAMIA-SBIA. vol. 1952, pp. 43–52. Springer (2000)
9. Hatcher, E., Gospodnetic, O.: Lucene in Action. Manning Publications (2004)
10. He, D., Göker, A., Harper, D.: Combining Evidence for Automatic Web Session Identification. Information Processing & Management 38(5), 727–742 (2002)
11. Jansen, B.J., Spink, A., Blakely, C., Koshman, S.: Defining a Session on Web Search Engines. Journal of the American Society for Information Science and Technology 58(6), 862–871 (2007)
12. Kanawati, R., Jaczynski, M., Trousse, B., J-M, A.: Applying the Broadway Recommendation Computation Approach for Implementing a Query Refinement Service in the CBKB Meta-search Engine. In: Conférence Française sur le Raisonnement á Partir de Cas (RáPC'99) (1999)
13. Lenz, M., Ashley, K.: AAAI Workshop on Textual Case-Based Reasoning (1999), AAAI Technical Report WS-98-12
14. Liu, S.B.: Trends in Distributed Curatorial Technology to Manage Data Deluge in a Networked World. The European Journal for the Informatics Professional 11(4), 18–24 (2010)
15. Morris, M.R., Teevan, J.: Collaborative Search: Who, What, Where, When, Why, and How (Synthesis Lectures on Information Concepts, Retrieval, and Services). Morgan and Claypool Publishers (2010)
16. Plaza, E.: Semantics and Experience in the Future Web. In: ECCBR. pp. 44–58 (2008)
17. Qiu, F., Cho, J.: Automatic Identification of User Interest for Personalized Search. In: WWW 06: Proceedings of the 15th International Conference on the World Wide Web. pp. 727–736. ACM Press, New York, NY, USA (2006)
18. Rissland, E.L., Daniels, J.J.: A Hybrid CBR-IR Approach to Legal Information Retrieval. In: Proceedings of the 5th International Conference on Artificial Intelligence and Law. pp. 52–61. ACM Press (1995)
19. Saaya, Z., Smyth, B., Coyle, M., Briggs, P.: Recommending Case Bases: Applications in Social Web Search. In: Proceedings of 19th International Conference on Case-Based Reasoning, ICCBR 2011. pp. 274–288 (2011)
20. Smyth, B., Balfe, E., Freyne, J., Briggs, P., Coyle, M., Boydell, O.: Exploiting Query Repetition and Regularity in an Adaptive Community-Based Web Search Engine. User Model. User-Adapt. Interact. 14(5), 383–423 (2004)
21. Smyth, B., Briggs, P., Coyle, M., O'Mahony, M.: A Case-Based Perspective on Social Web Search. In: McGinty, L., Wilson, D. (eds.) Case-Based Reasoning Research and Development, Lecture Notes in Computer Science, vol. 5650, pp. 494–508. Springer Berlin / Heidelberg (2009)
22. Smyth, B., Champin, P.A.: The Experience Web: A Case-based Reasoning Perspective. In Grand Challenges for Reasoning from Experiences, Workshop at IJCAI'09 pp. 566–573 (2009)
23. Weber, R.O., Ashley, K.D., Bruninghaus, S.: Textual Case-based Reasoning. Knowledge Engineering Review 20(3), 255–260 (2005)