<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Demonstrating social search a la HeyStaks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors(s)</strong></td>
<td>Smyth, Barry; Coyle, Maurice; Briggs, Peter</td>
</tr>
<tr>
<td><strong>Publication date</strong></td>
<td>2009-10</td>
</tr>
<tr>
<td><strong>Conference details</strong></td>
<td>Presented at the 1st International Workshop on Recommendation-Based Industry Applications at The 3rd ACM Conference on Recommender Systems (RecSys 2009), October 2009, New York, NY, USA.</td>
</tr>
<tr>
<td><strong>Item record/more information</strong></td>
<td><a href="http://hdl.handle.net/10197/2289">http://hdl.handle.net/10197/2289</a></td>
</tr>
</tbody>
</table>
ABSTRACT
For all the success of mainstream search engines there are a number of opportunities for improving on the conventional Web search user experience. In this short paper we consider the default assumption that search is solitary in nature, an isolated interaction between individual user and search engine. We highlight the value of a more collaborative approach to Web search and briefly present a novel add-on for mainstream search engines: HeyStaks (www.heystaks.com). It is designed to provide a more collaborative search experience, one in which recommendation technologies play a central role, by learning from the search experiences of groups of searchers in order to provide targeted recommendations during future search sessions.

1. INTRODUCTION
The world of Web search is usually viewed as a solitary place. Although millions of searchers use services like Google and Yahoo everyday, their individual searches take place in isolation. Recently, researchers have begun to question the solitary nature of Web search, proposing a more collaborative search model in which groups of users can cooperate to search more effectively [2-6]. Recent work by [1] highlights the inherently collaborative nature of more general purpose Web search. For example, during a survey of just over 200 respondents, clear evidence for collaborative search behaviour emerged. More than 90% of respondents indicated that they frequently engaged in collaboration at the level of the search process. For example, 87% of respondents exhibited “back-seat searching” behaviours, where they watched over the shoulder of the searcher to suggest alternative queries. A further 30% of respondents engaged in search coordination activities, by using instant messaging to coordinate searches. Furthermore, 96% of users exhibited collaboration at the level of search products, that is, the results of searches. For example, 86% of respondents shared the results they had found during searches with others by email. Indeed almost 50% of respondents telephoned colleagues directly to share Web search results, while others prepared summary documents and/or Web pages in order to share results with others.

Thus, despite the absence of explicit collaboration features from mainstream search engines there is clear evidence that users implicitly engage in many different forms of collaboration as they search, although, as reported by [1], these collaboration “work-arounds” are often frustrating and inefficient. Naturally, this has motivated researchers to consider how different types of collaboration might be supported by future editions of search engines. In this paper we describe the HeyStaks collaborative Web search service (www.heystaks.com) which has been designed to work with mainstream search engines such as Google, and which has recently been deployed online. HeyStaks takes the form of a browser toolbar to allow users to capture and share their search experiences with other users and in so doing facilitates the creation of search communities. In turn, members of these search communities benefit from recommendations that are derived from the activities of other community members. As a short paper our aim is not to provide an in-depth description of the HeyStaks system or its deployment. Rather, we wish to highlight its core recommendation features and show how a recommendation perspective has helped to shape a novel solution to some of the problems facing mainstream search engines. During the course of this paper we will refer the reader to related work that will provide additional technical details and evaluation results.

2. HEYSTAKS: A SEARCH UTILITY
In designing HeyStaks our primary goal is to provide social Web search enhancements, while at the same time allowing searchers to continue to use their favourite search engine. As such, a key component of the HeyStaks architecture is a browser toolbar that permits tight integration with search engines such as Google, allowing searchers to search as normal while providing a more collaborative search experience via targeted recommendations. In this section we will outline the basic HeyStaks system architecture and summarize how result recommendations are made during search. In addition we will make this discussion more concrete by briefly summarizing a worked example of HeyStaks in action.

2.1 System Architecture
HeyStaks adds two important collaboration features to any mainstream search engine. First, it allows users to create search staks, as a type of folder for their search experiences at search time. Staks can be shared with others so
that their own searches will also be added to the stak. Second, HeyStaks uses staks to generate recommendations that are added to the underlying search results that come from the mainstream search engine. These recommendations are results that stak members have previously found to be relevant for similar queries and help the searcher to discover results that friends or colleagues have found interesting, results that may otherwise be buried deep within Google’s default result-list.

As per Fig. 1, HeyStaks takes the form of two basic components: a client-side browser toolbar and a back-end server. The toolbar allows users to create and share staks and provides a range of ancillary services, such as the ability to tag or vote for pages. The toolbar also captures search result click-thrus and manages the integration of HeyStaks recommendations with the default result-list. The back-end server manages the individual stak indexes (indexing individual pages against query/tag terms and positive/negative votes), the stak database (stak titles, members, descriptions, status, etc.), the HeyStaks social networking service and, of course, the recommendation engine. In the following sections we will briefly outline the basic operation of HeyStaks and then focus on some of the detail behind the recommendation engine.

2.2 A Worked Example

To make HeyStaks more concrete it is useful to consider a worked example. With this in mind, consider the scenario where the leader of a recommender systems research group wishes to harness the search knowledge of his/her group to help other group members, particularly new researchers, to search more productively.

This is the scenario illustrated in Figure 2. To begin with, the group leader creates a new stak by selecting the “Create a New Stak” option from the “Staks” menu in the HeyStaks toolbar. As per Figure 2(a), creating a stak is a straightforward process: the stak creator needs to provide a stak name and some helpful description information; the stak can be configured to be public (anyone can join) or private (invitation only); and the creator can invite initial members by providing their email addresses. In this case the user creates the public RecSys stak and invites a group of researchers via the postgrads@clarity-centre.group-email address. If the researchers accept this invitation then the RecSys stak will be added to their HeyStaks toolbar.

At search time, HeyStaks users can select an active stak from their toolbar to provide a context for their search. For example, in Figure 2(b) the searcher has selected the RecSys stak in a search for “collaborative filtering” and the result list returned by Google has been augmented by HeyStaks promotions. In this case the top 3 results have been promoted by HeyStaks because they have each been found to be relevant to stak members, either during searches for similar queries or through their tagging activities. In addition to these primary recommendations, RecSys can also make a larger set of additional recommendations available. These may be drawn from the RecSys stak or indeed from other staks that the user has joined; in this case, HeyStaks has found additional recommendations from the RecSys stak and also from the user’s personal My Searches stak.

In this way, as stak members submit queries and select results, these search experiences are captured in the RecSys stak. In addition, as mentioned above, HeyStaks also allows users to more explicitly interact with search results and Web pages. For example: users can vote for (or against) particular results; users can email a page directly to another user without leaving the page or their search; and users can explicitly tag any page that they find to be interesting (see Figure 2(c)). This combination of implicit click-thru data and explicit voting, sharing, or tagging data permits staks to capture a variety of important interaction types which HeyStaks uses to infer the relevance of a given page to a given stak; see [7] for further detail.

Separately from the toolbar, HeyStaks users also benefit from the HeyStaks search portal, which provides a social networking service built around people’s search histories. For example, Figure 2(d) shows the portal page for the RecSys stak, which is available to all stak members. It presents an activity feed of recent search history and a query cloud that makes it easy for the user to find out about what others have been searching for. The search portal also provides users with a wide range of features such as stak maintenance (e.g., editing, moving, copying results in staks and between staks), various search and filtering tools (see Figure 2(e)), and a variety of features to manage their own search profiles and find new search partners.

2.3 The HeyStaks Recommendation Engine

In HeyStaks each search stak \( (S) \) serves as a profile of the search activities of the stak members. Each stak is made up of a set of result pages \( (S = \{p_1, \ldots, p_k\}) \) and each page is anonymously associated with a number of implicit and explicit interest indicators, including the total number of times a result has been selected \( (sel) \), the query terms \( (q_1, \ldots, q_n) \) that led to its selection, the number of times a result has been tagged \( (tag) \), the terms used to tag it \( (t_1, \ldots, t_m) \), the votes it has received \( (v^+, v^-) \), and the number of people it has been shared with \( (share) \) as indicated by Eq. 1.

\[
p_i^S = \{q_1, \ldots, q_n, t_1, \ldots, t_m, v^+, v^-, sel, tag, share\} \tag{1}
\]

In this way, each page is associated with a set of term data (query terms and/or tag terms) and a set of usage data (the selection, tag, share, and voting count). The term data is represented as a Lucene \((\text{lucene.apache.org})\) index, with
a) The user creates a RecSys stak as a repository for searches related to recommender systems work. Sharing the stak with research students allows others to contribute to, and benefit from, its growing search knowledge.

b) Now, any relevant Google searches will result in recommendations from the RecSys stak.

c) As users search and browse they can vote on or tag pages directly from the HeyStaks toolbar, ensuring that these pages are added to an active stak.

d) The HeyStaks search portal allows users to keep up-to-date with stak activity, maintain/curate search staks, as well as providing a range of functionality to help users to find relevant staks to join, discover new content etc.

e) Users can search for new staks to join, given an information need. Here we see how a search for ‘collaborative filtering’ has revealed a number of relevant staks, which the user can join or share with others.
each page indexed under its associated query and tag terms, and provides the basis for retrieving and ranking promotion candidates. The usage data provides an additional source of evidence that can be used to filter results and to generate a final set of recommendations. At search time, recommendations are produced in a number of stages: first, relevant results are retrieved and ranked from the stack index; next, these promotion candidates are filtered based on the usage evidence to eliminate noisy recommendations; and, finally, the remaining results are added to the Google result-list according to a set of recommendation rules.

Retrieval & Ranking. Briefly, there are two types of promotion candidates: primary promotions are results that come from the active stack \( S_i \); whereas secondary promotions come from other stacks in the searcher’s stack-list. To generate these promotion candidates, the HeyStaks server uses the current query \( q_i \) as a probe into each stack index, \( S_i \), to identify a set of relevant stack pages \( P(S_i, q_i) \). Each candidate page, \( p_i \), is scored using a \( TF*IDF \)-based retrieval function as per Equation 2, which serves as the basis for an initial recommendation ranking.

\[
score(q_i, p_i) = \sum_{t \in q_i} tf(t, p_i) \cdot idf(t)^2 \quad (2)
\]

Evidence-Based Filtering. Staks are inevitably noisy, in the sense that they will frequently contain pages that are not on topic. As a result, the retrieval and ranking stage may select pages that are not strictly relevant to the current query context. To avoid making spurious recommendations HeyStaks employs an evidence filter, which uses a variety of threshold models to evaluate the relevance of a particular result, in terms of its usage evidence; tagging evidence is considered more important than voting, which in turn is more important than implicit selection evidence. For example, pages that have only been selected once, by a single stack member, are not automatically considered for recommendation and, all other things being equal, will be filtered out at this stage. In turn, pages that have received a high proportion of negative votes will also be eliminated. The precise details of this model are beyond the scope of this paper but suffice it to say that any results which do not meet the necessary evidence thresholds are eliminated from further consideration.

Recommendation Rules. After evidence pruning we are left with revised primary and secondary promotions and the final task is to add these qualified recommendations to the Google result-list. HeyStaks uses a number of different recommendation rules to determine how and where a promotion should be added. Once again, space restrictions prevent a detailed account of this component but, for example, the top 3 primary promotions are always added to the top of the Google result-list and labelled using the HeyStaks promotion icons. If a remaining primary promotion is also in the default Google result-list then this is labeled in place. If there are still remaining primary promotions then these are added to the secondary promotion list, which is sorted according to HeyStaks relevance values. These recommendations are then added to the Google result-list as an optional, expandable list of recommendations.

3. DISCUSSION

HeyStaks is designed to help users to collaborate during Web search tasks and, importantly, it succeeds in integrating collaborative recommendation techniques with mainstream search engines. The HeyStaks system has recently moved into public-beta and during this time approximately 500 users have registered, leading to the creation and sharing of thousands of search stacks. This has provided an opportunity to better understand the usage patterns of HeyStaks users and the extent to which users benefit from HeyStaks recommendations. For example, on average users created approximately 3 stacks each and joined between 1 and 2 further stacks. Indeed, we found the sharing of stacks to be common within the user-base with 70% of users sharing stacks with other users. Moreover, users frequently responded positively to HeyStaks’ recommendations. For example, 85% of users selected recommendations that were the result of the activities of other stack members.

We have only scratched the surface of the HeyStaks system in this short paper. Our aim has been to introduce HeyStaks, rather than provide a deep technical analysis and evaluation. Nevertheless we have described the system architecture and recommendation components and provided a detailed use-case of the system in operation. It is worth noting that our focus to date has been on just one recommendation opportunity in HeyStaks, namely the recommendation of results at search time, and it is interesting to note a number of further recommendation opportunities and challenges. For example, in recent work we have considered the problem of stack recommendation rather than result recommendation, to describe how a relevant stack can be suggested to a user at search time from their stack-list. Indeed, more generally, it is interesting to consider the challenge of recommending stacks to the user at search time that they have not yet discovered, to help them benefit from the search knowledge of more distant communities.

4. ACKNOWLEDGEMENTS

Based on works supported by Science Foundation Ireland, Grant No. 07/CE/I1147 and HeyStaks Technologies Ltd.

5. REFERENCES