Research Repository UCD

Title	Towards a reputation-based model of social web search
Authors(s)	McNally, Kevin, O'Mahony, Michael P., Smyth, Barry, Coyle, Maurice, Briggs, Peter
Publication date	2010-02-07
Publication information	McNally, Kevin, Michael P. O'Mahony, Barry Smyth, Maurice Coyle, and Peter Briggs. "Towards a Reputation-Based Model of Social Web Search." Association for Computing Machinery, 2010.
Conference details	2010 International Conference on Intelligent User Interfaces (IUI 2010), Hong Kong, China, 7-10 February 2010
Publisher	Association for Computing Machinery
Item record/more information	http://hdl.handle.net/10197/2034
Publisher's version (DOI)	10.1145/1719970.1719996

Downloaded 2023-04-07T10:52:29Z

The UCD community has made this article openly available. Please share how this access benefits you. Your story matters! (@ucd_oa)

© Some rights reserved. For more information

Towards a Reputation-based Model of Social Web Search

Kevin McNally, Michael P. O'Mahony, Barry Smyth, Maurice Coyle, Peter Briggs
CLARITY: Centre for Sensor Web Technologies
School of Computer Science and Informatics
University College Dublin, Ireland
{firstname.lastname}@ucd.ie

ABSTRACT

While web search tasks are often inherently collaborative in nature, many search engines do not explicitly support collaboration during search. In this paper, we describe *HeyStaks* (www.heystaks.com), a system that provides a novel approach to collaborative web search. Designed to work with mainstream search engines such as Google, HeyStaks supports searchers by harnessing the experiences of others as the basis for result recommendations. Moreover, a key contribution of our work is to propose a reputation system for HeyStaks to model the value of individual searchers from a result recommendation perspective. In particular, we propose an algorithm to calculate reputation directly from user search activity and we provide encouraging results for our approach based on a preliminary analysis of user activity and reputation scores across a sample of HeyStaks users.

Author Keywords

Collaborative Web Search, Reputation Model, HeyStaks

ACM Classification Keywords

H.4.0 Information Systems Applications: General

General Terms

Algorithms, Experimentation, Security

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 [10, 11, 12, 13, 17]. Indeed, recent work by [4] highlights the inherently collaborative nature of more general purpose web search. 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 these collaboration “work-arounds” are often frustrating and inefficient [4]. Naturally, this has motivated researchers to consider how different types of collaboration might be supported by future editions of search engines. *HeyStaks* is one such model of collaborative web search (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.

During the initial trials of HeyStaks it has become clear that different users engage in, and benefit from, different degrees of search collaboration [19, 20]. For example, clear *search leaders* and *search followers* often emerge, the former being consistently first to contribute search knowledge for the latter to consume in the form of recommendations. In this paper, we consider the notion of *reputation* as a measure of how reliable a searcher is when it comes to the production of useful search knowledge. For example, if a particular searcher contributes search knowledge that is frequently selected by others during future searches, then the reputation of that searcher should be credited. In this paper, we describe one such reputation model and discuss the results of a preliminary evaluation across a subset of collaborating users. First, however, we review recent work in the area of collaborative information retrieval and summarise the HeyStaks system that forms the basis for this work.

COLLABORATIVE INFORMATION RETRIEVAL

Collaborative information retrieval research takes a fresh look at information retrieval and web search, which highlights the potential for collaboration between searchers during extended search tasks. Recent work by [4] 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. Some 30% of respondents engaged in search coordination activities, by using instant messaging to coordinate searches. Fur-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

IUI'10, February 7–10, 2010, Hong Kong, China.

Copyright 2010 ACM 978-1-60558-515-4/10/02...\$10.00.

thermore, 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 [4], these collaboration “work-arounds” are often frustrating and inefficient. This has motivated researchers to consider how different types of collaboration might be supported by future editions of search engines. The resulting approaches to *collaborative information retrieval* can be distinguished in terms of two dimensions – *time* and *place*. In terms of the former, collaborative search systems can be designed to support *synchronous* or *asynchronous* collaborative search. And in terms of the latter, systems can be designed to support either *co-located* or *remote* forms of collaborative search.

Co-located systems offer a collaborative search experience for multiple searchers at a single location, often via a single PC [1] or, more recently, by taking advantage of computing devices that are more naturally collaborative, such as table-top computing environments [16]. In contrast, remote approaches allow searchers to perform their searches at different locations across multiple devices [5, 6, 20]. While co-located systems enjoy the obvious benefit of an increased facility for direct collaboration that is enabled by the face-to-face nature of co-located search, remote services offer a greater opportunity for collaborative search.

Synchronous approaches are often characterised by systems that broadcast a “call to search” in which specific participants are requested to engage in a well-defined search task for a well defined period of time [15]. In contrast, asynchronous approaches are characterised by less well-defined, ad-hoc search tasks and provide for a more open-ended approach to collaboration in which different searchers contribute to an evolving search session over an extended period of time [5, 18]. In this paper we will focus on a community-based approach to collaborative web search in which the *asynchronous* search experiences of communities of like-minded *remote* searchers are harnessed to provide an improved search experience that is more responsive to the learned preferences of a community of searchers.

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

Figure 1. The HeyStaks system architecture and outline recommendation model.

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.

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. These staks can then 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 Figure 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.

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 search 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.org* 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 previous 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. 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 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 page to a given stak; see [19] for details.

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.

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 ($S = \{p_1, \dots, p_k\}$) 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 ($q_1, \dots, q_n$) that led to its selection, the number of times a result has been tagged (*tag*), the terms used to tag it ($t_1, \dots, t_m$), the votes it has received (v^+, v^-), and the number of people it has been shared with (*share*) as indicated by Eq. 1.

$$p_i^S = \{q_1, \dots, q_n, t_1, \dots, t_m, v^+, v^-, sel, tag, share\} \quad (1)$$

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 (*lucene.apache.org*) index, with 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 stak 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 stak S_t ; whereas *secondary promotions* come from other staks in the searcher’s stak-list. To generate these promotion candidates, the HeyStaks server uses the current query, q_t , as a probe into each stak index, S_i , to identify a set of relevant stak pages, $P(S_i, q_t)$.

Each candidate page, p , is scored against the query, q_t , using a *term frequency-inverse document frequency* (TF-IDF) based retrieval function. TF-IDF is a well-known weighting scheme from the field of information retrieval and is a measure used to weight the importance of a term within a collection of documents [14]. The value of this weight is proportionate to the frequency of the term in a particular document, but is offset by its frequency across the entire corpus. Such an approach serves as the basis for an initial recommendation ranking in HeyStaks, as per Equation 2.

$$score(q_t, p) = \sum_{t \in q_t} tf(t \in p) \bullet idf(t)^2 \quad (2)$$

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*. This filter uses a variety of threshold models to evaluate the relevance of a particular result in terms of its usage evidence. For example,

Figure 2. HeyStaks in action: a) stak creation; b) result recommendations; c) tagging a web page; d) stak activity on the HeyStaks portal; e) finding new staks.

tagging evidence is considered more important than voting, which in turn is more important than implicit selection evidence. Pages that have only been selected once, by a single stak member, are not automatically considered for recommendation by HeyStaks 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.

Summary 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. In preceding sections, we have provided an overview of the various functionality that HeyStaks provides and have discussed the ranking, filtering and recommendation techniques that are used to make result promotions. Further details on the precise techniques employed can be found in previous research [2, 18]. 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 staks. In the next section, we introduce a reputation model for HeyStaks users and we show how this model can be used to identify the search leaders in a community and also how user reputation can be used to enhance the ranking of result promotions.

A REPUTATION MODEL FOR SOCIAL SEARCH

As described in the previous section, the many and varied different types of activities that a user can perform (click-thrus, tagging, voting, sharing) on a web page are ultimately combined and leveraged by HeyStaks to make recommendations at search time. And, while the recommendation algorithm used differentially weights different activity types (so that tagging, for example, is considered a more reliably indicator of interest than a simple result click-thru), the source of the activity (that is, the user performing the activity) is not considered explicitly. Intuitively, we might expect that some users are more experienced searchers and, as such, perhaps their activities should be considered as more reliable at rec-

ommendation time, so that promotion candidates that hail from the activities of very experienced users might be considered ahead of candidates that come from the activity of less experienced users. This is particularly important given the potential for malicious users to disrupt stak quality by introducing dubious results to a stak.

In this section then, we describe how user activities in HeyStaks can be harnessed to generate a computational model of user reputation, based on the collaboration events that naturally occur between HeyStaks users who share their search experiences. We describe an algorithm for maintaining an up-to-date reputation model at search time and go on to propose a simple mechanism for incorporating reputation into the HeyStaks result recommendation subsystem.

From Activities to Reputation

It seems natural that the reputation of a searcher should be linked to the search knowledge that they contribute to HeyStaks. In simple terms this search knowledge is based on the creation and sharing of search staks and, ultimately, the web pages that are added to these staks according to a variety of different types of user activities, which include:

- *Click-thrus (Result Selections)* – that is, a user selects a search result (whether *organic* or *promoted*);
- *Voting* – that is, a user positively votes on a given search result or the current web page;
- *Sharing* – that is, a user chooses to share a specific search result or web page with another user (via email or by posting to their Facebook Wall etc.);
- *Tagging/Commenting* – that is, the user chooses to tag and/or comment on a particular result or web page.

Each of these activities results in the creation of new search knowledge. If the target page is new to a stak, then its selection, sharing, voting, or tagging will cause it to be added to the stak for the first time. If the page is already represented, as a result of an earlier activity, then the page's stak record will be updated to reflect the additional activity. As mentioned previously, not all of these activities are equal with respect to their reliability as indicators of relevance/quality. For example, the act of selecting a search result (a click-thru) is considered to be an *implicit* activity, which may or may not indicate that the user views the page to be relevant; for example, if the user quickly dismissed the selected page and returns to searching then it is unlikely that the page was considered to be particularly useful or relevant. In contrast, the other activities (sharing, voting, tagging) are *explicit* and thus tend to be more reliable indicators of page relevance.

What then is the relationship between search activity and searcher reputation? Under the heading of "*more search knowledge is better than less search knowledge*" it might make sense to model reputation as a direct function of the sheer volume of activity that a given searcher engages in. This would be a mistake. For a start, just because a user is creating a lot of search knowledge, by adding many pages

to search staks, it does not mean that this new knowledge is useful, especially to others. On the contrary, one of the major concerns in any social recommender is the potential for misuse through the actions of malicious users, a problem that would no doubt be exacerbated by valuing the contribution of very ‘productive’ malicious users.

Ultimately, in a social media context, reputation is a form of *incentive*. It allows HeyStaks to communicate the value of a user’s contributions to that user, and potentially to others, and this can help significantly to drive further contributions [8, 9]. (Related to this is the concept of trust in recommender systems and social networks [3, 7] where, for example, the accumulation of trust scores can motivate users to enhance the quantity and quality of their contributions.) But like any incentive, reputation can be *gamed* and thus it is vitally important that the incentive is tightly coupled to the sort of behaviour that benefits the system and its users as a whole. A reputation model that is the sum of all user activities does not meet this requirement since it is not necessarily to anyone’s benefit to create a system that is measured simply by the volume of its search knowledge. Instead, it is the quality of this search knowledge that is important, and so our model of reputation must model search knowledge quality. The long-term value of HeyStaks as a social search service depends critically on the ability of users to benefit from its quality search knowledge and if, for example, all of the best search experiences are tied up in private staks and never shared, then this long-term value will be greatly diminished.

Reputation as Collaboration

Thus, our model of reputation must recognise the *quality* of *shared* search knowledge. Fortunately there is a way to capture this notion in a manner that serves to incentivise users to behave in just the right way to grow long-term value for all. The key idea is that, ultimately, the quality of shared search knowledge can be estimated by looking at the frequency of *search collaborations* within HeyStaks.

If HeyStaks recommends a result to a searcher, and the searcher chooses to act on (select, tag, vote on or share) this result, then we can view this as a single instance of search collaboration. The current searcher who chooses to act on the recommendation is known as the *consumer* and, in the simple case, the original searcher whose earlier action on this result caused it to be added to the search stak is known as the *producer*. In other words, the producer created search knowledge that was deemed to be useful enough for the consumer to act upon it. And the basic idea behind our reputation model is that this act of implicit collaboration between producer and consumer confers a *unit of reputation* on the producer (Figure 3). If a given user is a regular producer of search knowledge that is frequently recommended to, and acted on by, many other users, then this producer will enjoy a high reputation score. Moreover, if users create lots of staks and share these staks with many other users, or simply join staks that have been created by others, then they create an opportunity for more collaboration; and if users contribute good search knowledge to shared staks then their reputation score will benefit from the realisation of these frequent

collaboration opportunities. In this way, this collaboration-based model of reputation is incentivizing users not just to create search knowledge but also to share it with others.

Figure 3. Producer (P) and consumer (C) collaboration: C selects page p_i , which has been recommended to C based on P ’s previous activity. In turn, C confers reputation on P .

The conferral of reputation by a consumer is a little more complicated than just described because, in the general case, at the time when the consumer acted on the promoted result, there may have been a number of different producers who each contributed part of the search knowledge that caused this result to be promoted. An original producer may have been the first to select the result in question, but subsequent users may have selected it for different queries, or they may have voted on it or tagged it or shared it with others independently of its other producers. In the case of our reputation model we share the unit of reputation between the other producers. So, if at time t , when the consumer acts on a promoted result, we can identify k producers then the reputation score of each of these producers is incremented by $1/k$.

An Example

To illustrate our user reputation model, consider the simple scenario as depicted in Figure 4. Here, the activity of four users, $\{u_1, \dots, u_4\}$, with respect to a single search result page p is shown at four points in time t_i , where $t_4 > t_3 > t_2 > t_1$. Further, assume that all four users are members of a particular stak S , which is currently the active stak for each of these users. The sequence of events at each time step t_i is as follows:

- t_1 : User u_1 organically selects page p for some search query q , causing page p to be added to stak S .
- t_2 : User u_2 selects page p , which has been promoted by HeyStaks, for a search query that is related to q . Since user u_1 is the only user to have previously selected page p in stak S , we say that user u_1 (the *producer*) has *promoted* page p to user u_2 (the *consumer*). Consequently, user u_2 assigns a *reputation* score of 1 to user u_1 .
- t_3 : User u_3 organically selects page p for an unrelated search query q' . This time, page p is not promoted by HeyStaks and hence no reputation is assigned by user u_3 to any of the other users.
- t_4 : Finally, user u_4 selects page p , which has been promoted by HeyStaks, for a search query that is again related to q . Since users u_1 , u_2 and u_3 have all previously selected (either organically or by promotion) page p , on this occasion reputation is assigned by user u_4 to each of these users. Thus, in Figure 4, the reputation score is distributed equally among the three users, such that each user receives a score of $1/3$.

Figure 4. Simple example of user reputation calculations in HeyStaks

At the end of the time period, overall user reputation is calculated by simply summing the individual reputation components that each user has received. For example, in the above scenario, the overall reputation scores for users u_1 , u_2 , u_3 and u_4 are $4/3$, $1/3$, $1/3$ and 0 , respectively.

The complete user reputation algorithm is given in Figure 5. For the purposes of simplicity, this algorithm shown is one suitable for offline execution. The algorithm can be readily modified such that user reputation scores are updated in real time when new activities are performed by users. In future work, we plan on integrating such a version into the HeyStaks application.

The algorithm takes as input a temporally ordered set of user activities $\mathcal{A}$ which are retrieved from the HeyStaks activity feed (e.g. Figure 2(d)). Each entry $a \in \mathcal{A}$ is a tuple $\langle u, p, t, S, type \rangle$, where $a.u$ is the user who performed the activity, $a.p$ is the associated result page, $a.t$ is the time when the activity occurred, $a.S$ is the active stak at the time of the activity and $a.type$ indicates whether or not the activity relates to a HeyStaks promotion. In addition, the set of all staks $\mathcal{S}$ and the current (previously calculated) set of user reputation scores $\mathcal{R}$ are provided as a starting point.

For each promotion activity $a \in \mathcal{A}$ (line 3), the set of staks S_c that the current user u_c is a member of is retrieved (line 8). Then, the set of prior activities relating to the current page p_c , in any of the staks in S_c , is determined (line 9) and the users who performed these activities are identified (line 10). Finally, a unit of reputation is distributed equally among these users and added to their existing reputation score (lines 12–14). This process continues until all activities are processed and the array $\mathcal{R}$, which contains each user’s updated reputation score, is returned.

Result Promotion

We now consider how user reputation can be employed to influence the ranking of promoted results in HeyStaks. Currently, pages are selected for promotion as follows. For a given search query q_t submitted by a user u , a set of candidate pages, $\{p_1, p_2, \dots, p_k\}$, are identified for promotion

Input: Set $\mathcal{A}$ of user activity tuples $\langle u, p, t, S, type \rangle$, set $\mathcal{S}$ of all staks, array $\mathcal{R}$ of user reputation scores

Output: Updated array $\mathcal{R}$ of user reputation scores

```

1.  USERREPUTATION( $\mathcal{A}, \mathcal{S}, \mathcal{R}$ )
2.  begin
3.    foreach activity  $a \in \mathcal{A}$ 
4.      if  $a.type = \text{promotion}$ 
5.         $u_c \leftarrow a.u$ 
6.         $p_c \leftarrow a.p$ 
7.         $t_c \leftarrow a.t$ 
8.         $S_c \leftarrow \text{staks}(\mathcal{S}, u_c)$ 
9.         $A \leftarrow \{a' \in \mathcal{A} : a'.p = p_c \ \& \ a'.S \in S_c \ \& \ a'.t < t_c\}$ 
10.        $U \leftarrow \{a'.u : a' \in A\} - \{u_c\}$ 
11.
12.       foreach  $u \in U$ 
13.          $\mathcal{R}[u] \leftarrow \mathcal{R}[u] + 1/|U|$ 
14.       end
15.     end
16.   end
17. end

```

Figure 5. User reputation algorithm

and a relevance score, $score(q_t, p_i)$, is computed for each page (Eqn. 2). These scores are then used to rank order candidate pages, and the pages with the highest scores are promoted to the user.

We propose to incorporate user reputation into the above ranking process as follows. Let $rep(p_i)$ denote the reputation accruing to candidate page p_i . Candidate pages can now be ranked according to:

$$rank(p_i) = w \times rep(p_i) + (1 - w) \times score(q_t, p_i), \quad (3)$$

where w lies in the interval $[0, 1]$. Higher values of w increase the influence of page reputation on overall rankings.

The reputation of a candidate page p_i at time t is calculated as follows. Let U be the set of users who had selected, voted up, shared or tagged page p_i prior to time t . Further, let $\mathcal{R}$ be an array containing the reputation scores for each user $u \in U$, where $\mathcal{R}$ is the (normalised) output of the user reputation algorithm described above. The reputation of page p_i is calculated as:

$$rep(p_i) = \frac{\sum_{u \in U} \mathcal{R}[u]}{|U|}. \quad (4)$$

Eqn. 4 assigns larger scores to candidate pages that have been previously selected by users with high user reputations. Thus, by incorporating page reputation into the HeyStaks’ page ranking process as per Eqn. 3, the relevance of promoted pages can be further enhanced.

PRELIMINARY EVALUATION

Ultimately it is our intent to evaluate this reputation model as an integrated component within HeyStaks as part of a long-term user trial. Accordingly it will be possible to determine just how important a role reputation can play when it comes to influencing recommendations and user engagement. For example, will a reputation-based recommenda-

tion model lead to improved recommendations that attract more frequent click-thrus? And would exposing reputation statistics to users help to deepen their engagement with the system, leading to more collaboration in the long-run?

Unfortunately, this level of integration is beyond the scope of this work. However, we do have an opportunity to evaluate the reputation model with respect to a limited user community in order to explore and better understand the relationship between users, their activities, and their reputation scores.

Dataset & Methodology

For this initial evaluation, we considered 26 HeyStaks users who have been using the system over the course of the last 9 months. These users were invited to try out the system and can be said to be typical of early adopters of new systems. As such, they may be more technically knowledgeable than the average user and thus our findings may not fully generalise to regular users. Nonetheless, our test group is typical of that used in many system trials (e.g. [20]) and useful findings and insights can be obtained from our study.

The activity data associated with our user group captures a wide range of information about user activity within HeyStaks, including stak creation, joining and sharing as well as page selection, tagging, voting and sharing. In total, some 20,472 individual activity records are included and these provide the basic input to the reputation model described above to form the basis of our evaluation. Note that we focus on these 26 users as the *holders* of reputation, but it is important to realise that they may have received this reputation from a wider set of users who do not form part of this test-group.

User Activity

Overall, user activity in HeyStaks is dominated by selection actions, where the user selects a particular search result, be it an organic result or a recommended/promoted result. Figure 6 presents the total number of selections, tags, votes and shares that have been performed by the 26 users. Selections are of course a natural type of search activity (approximately 90%) and so it is unsurprising that they dominate compared to the other activities such as tagging (3.5%), voting (1.6%), and sharing (4%).

Figure 6. User search activity versus count

Figure 7(a) shows summary statistics in relation to the total activity (i.e. selection, tagging, voting, sharing) carried out

by the 26 users. The median total activity across the users is 638 with the most active user performing as many as 3528 activities, while the least active user performed only 11 actions. This suggests a reasonable spread of activity across the 26 test users.

Another important issue to consider is the extent to which users engage in the sort of activities that ultimately facilitate search collaboration. Do they create and join staks, for example? How many other users are they connected with (their *network size*)? All other things being equal, if a user creates and joins many staks, then they are more likely to be connected to a greater number of other users and thus will benefit from a larger collaboration network to act as a source of recommendations. Figures 7(b) to 7(d) presents the summary statistics for the number of staks created and joined and the network size for the 26 test users.

We can see that the median number of staks created is 3 versus 8 staks joined. So the typical user tends to join more staks (which others have created) than they create. This is an important indication that users are recognising the potential value of other staks which other users have created and is a first step towards meaningful search collaboration. Once again there are some significant outliers with, at one extreme, a user creating 36 staks while another joined 49, whilst, at the other extreme, one user only created a single stak while another joined only 2. It is worth highlighting that there is a strong correlation (0.97) between the number of staks created and the number of staks joined.

The median network size – that is, the number of other users a given user is *connected* with by virtue of sharing staks – is 30 (this value includes other users who were not part of this particular analysis). Indeed one user is connected to 89 others while another is only connected to 11. These summary statistics suggest that not only are users engaged in a significant level of search activity within HeyStaks, they have also created the conditions (shared staks) for meaningful collaboration.

User Reputation

The results of applying the reputation model to the 26 users are shown in Figure 8, with users ordered by decreasing reputation score. Remember that the reputation score is basically a weighted sum of the number of times a particular user has contributed to the promotion of a result that has been subsequently been acted on (i.e. selected, shared etc.). If the user in question was the sole producer of the promotion then they gain a full unit of reputation, but more often than not they are partly responsible for the promotion and so only share in a fraction of the reputation along with the promotion's other producers.

The results in Figure 8 tell a tale of two user types. On the one hand, about 20% of the users (5 out of 26) have achieved reputation scores of 60 or higher, with one user achieving a reputation score in excess of 105. These are clearly users who are engaged in a significant amount of search collaboration as distinct from the other users (21 out of 26) who have

Figure 7. Maximum, median and minium scores for activity metrics across 26 users

helped to drive some, although a relatively small amount of, collaboration. These users have reputation scores between 1 and 25 and as such have proven to be less important when it comes to producing search knowledge that is of use to others (although no doubt these users are consuming promotions that others have produced). We view the smaller subset of high-reputation users as the *search leaders* within HeyStaks, while there is a larger subset of *search followers* who are more likely to consume than produce search knowledge.

Figure 8. Reputation scores for 26 users

Activity versus Reputation

Earlier in the paper, we cautioned against reputation models that reward users purely on the basis of the accumulation of some activity. Such models can be readily exploited to reward unproductive activity that does not contribute to the core value of a service in and of itself. In this paper we have strived to develop a reputation model that is closely linked with the type of activities that are likely to reward the good behaviour of users whose actions contribute to the long-term value of the system. In HeyStaks this long-term value is ultimately invested in the ability of the system to support meaningful collaboration between searchers.

Nonetheless, it is interesting to understand the strength of the relationship between factors such as activity, staks created/joined, network size and user reputation scores. This correlation information is presented in Figure 9 and it is interesting to note that there is a clear relationship between the factors and the user reputation score. Overall, the degree of user activity comes out on top compared to the number of staks created or joined and network size. This shows that

creating and joining staks may be necessary but it is not sufficient to drive reputation, at least in the absence of the contribution of the user, through their activity, to their search network's knowledge. Correlation with network size across users is comparatively low, suggesting that while having a large network does indeed provide a user with a good opportunity to collaborate, some users may prefer to work together in more tightly-knit search communities.

Figure 9. The correlation between activity, staks created/joined, network size and user reputation scores.

CONCLUSIONS

Even though mainstream search engines do not explicitly support collaboration during search, there is much evidence that many search tasks are inherently collaborative. One contribution of this paper is a description of a novel approach to collaborative web search that is fully compatible with mainstream web search engines. The HeyStaks system (www.heystaks.com) helps users to create search networks as a platform for search collaboration.

The main contribution of this paper, however, is the proposal of a reputation system for HeyStaks as a way to model the value of individual searchers, in terms of a reputation score, in order to weight their contributions during result recommendation. The key insight behind the proposed model is that the reputation of a user can be best measured by looking at how often the user is responsible for result recommendations that are ultimately selected. We have described how reputation can be calculated directly from user activity and we have provided some preliminary results based on an analysis of user activity and reputation scores across a sample of HeyStaks users. In summary the reputation model, while correlated with factors such as the number of staks

created/joined, the size of a users network, and the activity level of the user, is not dominated by any single factor and so should help to preserve the integrity of the model.

This reputation model will be especially important as a way to protect HeyStaks from malicious users who are motivated to game the system. While we accept that no system is totally foolproof, the HeyStaks system does provide a significant degree of protection against gaming. For example, search knowledge is partitioned into staks that have separate memberships, and this makes it difficult for a user to universally influence search results. HeyStaks also permits stak owners to curate their staks; owners can edit and delete stak contents and ban certain users if they are attempting to game a stak. Furthermore, the reputation algorithm confers reputation on a user u if and only if some other user v ($<> u$) selects a promotion that was derived from the search actions of u . Thus, if u is a spammer and contributes irrelevant or self-interested results to a stak, then these results are unlikely to be promoted and so u never benefits from a reputation increase.

Our paper is just a starting point for this work. The evaluation provided is of a preliminary nature, although it points in the right direction. Our next steps include a tight integration of the reputation system with the deployed HeyStaks system as the basis for a longer-term study that will focus on, for example, the following important issues: (1) does the reputation model lead to improved recommendations with higher click-thru rates; (2) does the availability of reputation statistics, on a user-by-user basis, help to provide users with useful feedback about their *search value* and does this deepen their engagement with HeyStaks and its search communities; (3) identifying the types of gaming activities and strategies that are likely to be carried out against HeyStaks and developing the reputation model to provide robustness against such activities.

ACKNOWLEDGEMENTS

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

REFERENCES

1. S. Amershi and M. R. Morris. Cosearch: a system for co-located collaborative web search. In *CHI*, pages 1647–1656, 2008.
2. O. Boydell and B. Smyth. Capturing community search expertise for personalized web search using snippet-indexes. In *CIKM*, pages 277–286, 2006.
3. U. Kuter and J. Golbeck. SUNNY: A new algorithm for trust inference in social networks, using probabilistic confidence models. In *AAAI*, pages 1377–1382, 2007.
4. M. R. Morris. A survey of collaborative web search practices. In *CHI*, pages 1657–1660, 2008.
5. M. R. Morris and E. Horvitz. S^3 : Storable, shareable search. In *INTERACT (1)*, pages 120–123, 2007.
6. M. R. Morris and E. Horvitz. Searchtogether: an interface for collaborative web search. In *UIST*, pages 3–12, 2007.
7. J. O’Donovan and B. Smyth. Trust in recommender systems. In *IUI*, pages 167–174, 2005.
8. J. Preece and B. Shneiderman. The reader to leader framework: Motivating technology-mediated social participation. *AIS Trans. on Human-Computer Interaction*, 1(1):13–32, 2009.
9. A. M. Rashid, K. Ling, R. D. Tassone, P. Resnick, R. Kraut, and J. Riedl. Motivating participation by displaying the value of contribution. In *CHI*, pages 955–958, 2006.
10. M. C. Reddy and P. Dourish. A finger on the pulse: temporal rhythms and information seeking in medical work. In *CSCW*, pages 344–353, 2002.
11. M. C. Reddy, P. Dourish, and W. Pratt. Coordinating heterogeneous work: Information and representation in medical care. In *ECSCW*, pages 239–258, 2001.
12. M. C. Reddy and B. J. Jansen. A model for understanding collaborative information behavior in context: A study of two healthcare teams. *Inf. Process. Manage.*, 44(1):256–273, 2008.
13. M. C. Reddy and P. R. Spence. Collaborative information seeking: A field study of a multidisciplinary patient care team. *Inf. Process. Manage.*, 44(1):242–255, 2008.
14. G. Salton and M. J. McGill. *Introduction to modern information retrieval*. McGraw-Hill, 1983.
15. A. F. Smeaton, C. Foley, D. Byrne, and G. J. F. Jones. ibingo mobile collaborative search. In *CIVR*, pages 547–548, 2008.
16. A. F. Smeaton, H. Lee, C. Foley, and S. McGivney. Collaborative video searching on a tabletop. *Multimedia Syst.*, 12(4-5):375–391, 2007.
17. B. Smyth. A community-based approach to personalizing web search. *IEEE Computer*, 40(8):42–50, 2007.
18. B. Smyth, E. Balfe, J. Freyne, P. Briggs, M. Coyle, and O. Boydell. Exploiting query repetition and regularity in an adaptive community-based web search engine. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, 14(5):383–423, 2004.
19. B. Smyth, P. Briggs, M. Coyle, and M. P. O’Mahony. A case-based perspective on social web search. In *ICCBR*, 2009.
20. B. Smyth, P. Briggs, M. Coyle, and M. P. O’Mahony. Google? shared! a case-study in social search. In *User Modeling, Adaptation and Personalization*. Springer-Verlag, June 2009.