A Case Study of Collaboration and Reputation in Social Web Search

KEVIN MCNALLY
MICHAEL P. O'MAHONY
BARRY SMYTH
PETER BRIGGS
and
MAURICE COYLE
CLARITY: Centre For Web Sensor Technologies, University College Dublin, Dublin, Ireland

1. INTRODUCTION

The scale of the Web and the heterogeneous nature of its content [Signorini and Gulli 2005] introduces many significant information discovery challenges. For all of the recent developments in search engine technologies, modern search engines continue to struggle when it comes to providing users with fast and efficient access to information. For example, recent studies have highlighted how even today's leading search engines fail to satisfy 50% of user queries [Smyth et al. 2005]. Part of the problem rests with the searchers themselves: with an average of only 2-3 terms [Lawrence and Giles 1998; Spink and Jansen 2004], the typical Web search query is often vague with respect to the searcher's true intentions or information needs [Song et al. 2007]. Moreover, searchers sometimes choose query terms that are not well represented in the page that they are seeking and so simply increasing the length of queries will not necessarily improve search performance.

Two promising and powerful new ideas in web search are *personalization* and *collaboration*. Personalization questions the *one-size-fits-all* nature of mainstream web search — two different users with the same query will receive the same result-list, despite their different preferences — and argues that web search needs to become more *personalized* so that the implicit needs and preferences of searchers can be accommodated [Chang et al. 2000; Chirita et al. 2004; Granka et al. 2004; Speretta and Gauch 2005; Asnicar and Tasso 1997; Ma et al. 2007; Makris et al. 2007; Zhou et al. 2006; Chirita et al. 2005; Pretschner and Gauch 1999; Shen et al. 2005; Budzik and Hammond 2000; Finkelstein et al. 2001].

This paper focuses on the second idea, that of collaboration. In the main, web search takes the form of an isolated interaction between lone searcher and search engine. Recently, however, there has been considerable interest in the potential for web search to evolve to become a more *social* activity [Morris et al. 2010; Golovchinsky et al. 2009; Evans et al. 2010; Evans and Chi 2009], whereby the search efforts of a user might be influenced by their social graph or the searches of others, potentially leading to a more *collaborative* model of search. In the broadest sense the idea of *social search* is one that tries to unify two distinctive information

discovery worlds: the traditional world of web search and the information sharing world of social networks. Only a few years ago, by and large, the majority of people located information of interest through their favourite mainstream search engine. But recently there has been a very noticeable change in how many web users satisfy their information needs. For example, recent statistics from Twitter claim that its users are explicitly searching tweet content 24 billion times per month[1] as compared to approximately 88 billion queries per month for Google. Similarly, at the time of writing, Facebook's own statistics highlight how its users are sharing upwards of 30 billion items of content every month.[2] Many of these items of content would have previously been located through mainstream search engines. Instead, today, they are being accessed via our social networks and, in terms of raw volume of information seeking activity, the social networks are now beginning to compete with mainstream search engines.

This shift in our information discovery habits has lead to an explosion in the number and variety of new social-search type services — all of which can influence our information discovery activities, bringing the world of web search and social networks even closer together (see Figure 1). In this context, social search can mean many things to many people. For some, social search is all about searching the real-time web (blogs and micro-blogs) à la the likes of InfoAxe, OneRiot, and Topsy. For others, social search is about indexing and filtering web content according to the online activities or opinions of users; see, for example, Mahalo (curated search categories), Scour (content indexed and filtered by real-time conversations) or the now-ended Wikia Search. For yet others, social search is about social bookmarking services (e.g. Delicious, XMarks, Twine), people search (e.g. Pipl, Nayms, Spock), or social news services (e.g., Digg, Reddit, Mixx).

Our aim is to make mainstream search engines more collaborative and to help people during routine search tasks by harnessing the recent search experiences of their friends and colleagues via their social networks. The focus of this paper is the HeyStaks search service (www.heystaks.com), which adds a layer of collaboration on top of mainstream search engines: so users continue to search as normal but benefit from a more collaborative/social search experience. The core HeyStaks system has been described in detail elsewhere [Smyth et al. 2009a; 2009b] and so we shall only review the HeyStaks approach in this paper. However, a key contribution of this paper is a detailed description of a recent live-user trial of HeyStaks in order to understand the usage and collaboration patterns of users and also the quality of HeyStaks' social recommendations relative to the organic results of mainstream search engines. In addition, a second contribution of this paper is a novel enhanced reputation model for HeyStaks, which has been developed in order to evaluate the reputation of individual searchers based on their search contributions. We go on to demonstrate how this reputation model can be used to further improve the quality of the HeyStaks recommendations, by prioritising those that originate from more reputable users.

[1] http://www.boygeniusreport.com/2010/07/07/twitter-handling-24-billion-search-queries-per-month/

[2] http://www.facebook.com/press/info.php?statistics

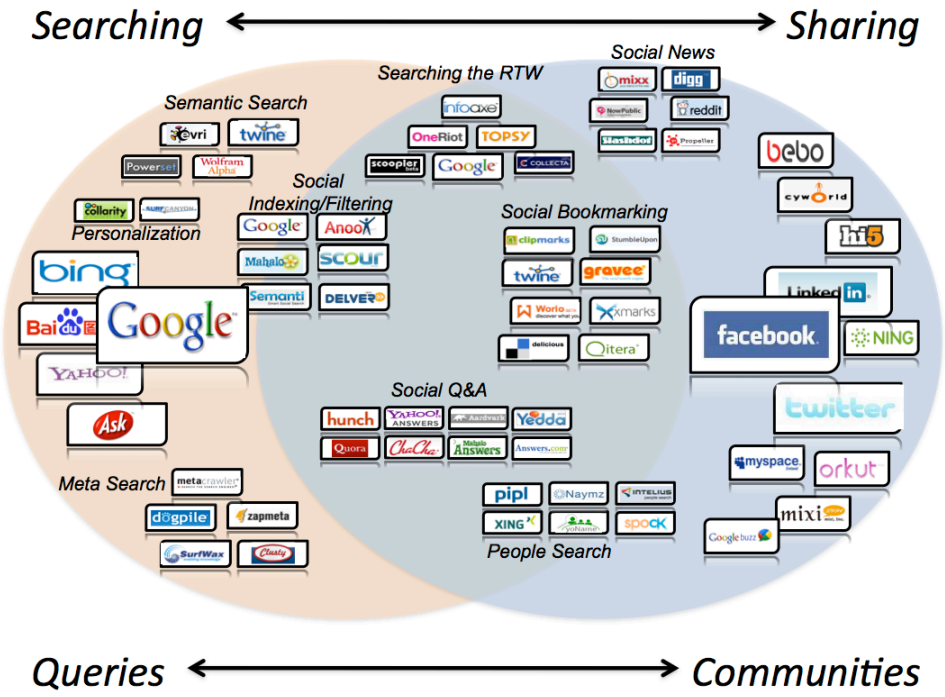


Fig. 1. Social search attempts to bridge the traditional, query-based world of web search with the information sharing world of social networks. A variety of social search and sharing services have emerged to help users harness their social networks in pursuit of more effective information discovery across a variety of application contexts. This figure lists a number of well-known services, both start-up and more mature, that have emerged to fill the gap between the mainstream search world such as Google, Yahoo and Bing, and the major social networks such as Facebook, Twitter and LinkedIn.

2. BACKGROUND

This paper focuses on discussing HeyStaks as a collaborative information retrieval technology, augmented by a reputation system based on the collaborations that implicitly take place between searchers in the HeyStaks social search utility. As such this background section covers recent, relevant work in the two broad areas of collaborative information retrieval and reputation systems.

2.1 Collaborative Information Retrieval

Approaches to *collaborative information retrieval* can be usefully distinguished in terms of two important dimensions, *time* — *synchronous* versus *asynchronous* search — and *place* — that is, *co-located* versus *remote* searchers. Co-located systems offer a collaborative search experience for multiple searchers at a single location, typically sharing a single PC [Amershi and Morris 2008; Smeaton et al. 2008], whereas remote approaches allow searchers to perform their searches at different locations across multiple devices [Morris and Horvitz 2007a; 2007b; Smyth et al. 2009b]. The former enjoy the obvious benefit of an increased faculty for direct

collaboration that is enabled by the face-to-face nature of co-located search, while the latter offer a greater opportunity for collaborative search. Alternatively, synchronous approaches are 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 [Smeaton et al. 2008]. 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 [Morris and Horvitz 2007a; Boydell and Smyth 2010].

A good example of the co-located, synchronous approach to collaborative web search is given by the work of Amershi and Morris [2008]. Their CoSearch system is designed to improve the search experience for co-located users where computing resources are limited; for example, a group of school children having access to a single PC. CoSearch is specifically designed to leverage peripheral devices that may be available (e.g. mobile phones, extra mice etc.) to facilitate distributed control and division of effort, while maintaining group awareness and communication. The purpose of CoSearch is to demonstrate the potential for productive collaborative web search in resource-limited environments. The focus is very much on dividing the search labour while maintaining communication between searchers, and live user studies speak to the success of CoSearch in this regard [Amershi and Morris 2008]. The work of Smeaton et al. [2007] is related in spirit to CoSearch but focuses on image search tasks using a table-top computing environment. Once again, preliminary studies speak to the potential for such an approach to improve overall search productivity and collaboration, at least in specific types of information access tasks. A variation on these forms of synchronous search activities is presented by Smeaton et al. [2008], where the use of mobile devices as the primary search device allows for a remote form of synchronous collaborative search. The iBingo system allows a group of users to collaborate on an image search task with each user using a iPod touch device as their primary search/feedback device (although conventional PCs appear to be just as applicable).

Remote search collaboration (whether asynchronous or synchronous) is the aim of SearchTogether, which allows groups of searchers to participate in extended shared search sessions as they search to locate information on particular topics [Morris and Horvitz 2007a]. The SearchTogether system allows users to create shared search sessions and invite other users to join in these sessions. Each searcher can independently search for information on a particular topic, but the system provides features to allow individual searchers to share what they find with other session members by recommending and commenting on specific results. SearchTogether supports synchronous collaborative search by allowing searchers to invite others to join in specific search tasks, allowing cooperating searchers to synchronously view the results of each others' searches via a split-screen style results interface. As with CoSearch above, one of the key design goals in SearchTogether is to support a division of labour in complex, open-ended search tasks. In addition, a key feature of the work is the ability to create a shared awareness among group members by reducing the overhead of search collaboration at the interface level. SearchTogether does this by including various features, from integrated messaging, query histories,

and recommendations arising out of recent searches.

The collaborative information retrieval systems we have so far examined have assumed the availability of an underlying search engine and provided a collaboration interface that effectively *imports* search results directly, allowing users to share these results. As noted by Pickens et al. [2008], one of the major limitations of these approaches is that collaboration is restricted to the interface, in the sense that while individual searchers are notified about the activities of collaborators, they must individually examine and interpret these activities in order to reconcile their own activities with their co-searchers. Consequently, work by Pickens et al. [2008] describes an approach to collaborative search that is more tightly integrated with the underlying search engine resource so that the operation of the search engine is itself influenced by the activities of collaborating searchers. For example, mediation techniques are used to prioritise, as yet, unseen documents, while query recommendation techniques are used to suggest alternative avenues for further search exploration.

HeyStaks has been designed to support collaborative web search tasks that are asynchronous and remote. Our objective is to tightly integrate this form of collaborative web search with mainstream search engines, which is a key point of differentiation with respect to previous collaborative search approaches as outlined above. An overview of the main components of the HeyStaks social search utility is given in Section 3.

2.2 Reputation Systems

Recently there has been considerable interest in *reputation systems* to provide a mechanism to evaluate user reputation and inter-user trust across a growing number of social web and e-commerce applications [Jøsang and Golbeck 2009; O'Donovan and Smyth 2005; 2006; Sabater and Sierra 2005; Resnick and Zeckhauser 2002; Resnick et al. 2000]. This work is, in part, motivated by the idea that an understanding of user reputation can serve as the basis for strategies to guard against malicious users [Lazzari 2010; Hoffman et al. 2009; Jøsang and Golbeck 2009]. Here, we present a brief review of the work that has been undertaken in this regard.

To begin, the reputation system used by eBay has been examined by Jøsang et al. [2007] and Resnick and Zeckhauser [2002]. Briefly, eBay elicits feedback from buyers and sellers regarding their interactions with each other, and that information is aggregated in order to calculate user reputation scores. The aim is to reward good behaviour on the site and to improve robustness by leveraging reputation to predict whether a vendor will honour future transactions. Resnick and Zeckhauser [2002] found that using information received directly from users to calculate reputation is not without its problems. Feedback is generally reciprocal; users almost always give positive feedback if they themselves had received positive feedback from the person they performed a transaction with. In many of these cases the information given is false, therefore reputation is not a reliable indicator of future vendor performance. Jøsang et al. [2007] confirms this, stating such systems require manual curation and protection from malicious users.

The work of O'Donovan and Smyth [2005] addresses reputation in recommender systems. Unlike conventional reputation systems like eBay's, reputation is not calculated by examining feedback received directly from users. Instead, the standard

collaborative filtering algorithm is modified to add a user-user trust score to compliment the normal profile or item-based similarity score, so that recommendation partners are chosen from those users that are not only similar to the target user, but who have also had a positive recommendation history with that user. O'Donovan and Smyth posit that reputation can be estimated by measuring the accuracy of a profile at making predictions over time. Using this metric average prediction error is improved by 22%.

Similar to O'Donovan and Smyth [2005], Massa and Avesani [2007] propose a reputation algorithm called *MoleTrust* that can be used to augment an existing collaborative filtering system. The mechanism calculates a "trust metric" similar to item-based similarity, which propagates across a network of content producers. This algorithm can be tuned to propagate over a specific depth across a social graph, meaning reputable users only have influence over a set of users of a known size. They find that *MoleTrust* can improve the accuracy of predictions made by a recommender system, even in cases where users have provided few ratings.

Other recent research has examined reputation systems employed in social networking platforms. Lazzari performed a case study of the professional social networking site Naymz [Lazzari 2010]. He warns that calculating reputation on a global level allows users who have interacted with only a small number of others to accrue a high degree of reputation, making the system vulnerable to malicious use. Similar to Jøsang et al. [2007], Lazzari [2010] suggests that vulnerability lies in the site itself, allowing malicious users to game the reputation system for their own ends. However, applying reputation globally affords malicious users influence over the entire system, which adds to its vulnerability. In Section 4, we present a computational model of user reputation which seeks to both protect the quality of HeyStaks recommendations in the face of malicious activity and to incentivise users to behave in a manner that promotes long-term value for all HeyStaks members.

3. HEYSTAKS: A SOCIAL 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. HeyStaks adds two basic 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, and the creator can invite initial members by providing their email addresses. Staks can be configured to be *public* (anyone can join) or *private* (invitation only). 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 shown in Figure 2, HeyStaks takes the form of two basic components: a client-side *browser toolbar* and a back-end *server*. The toolbar (see Figure 3) 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-

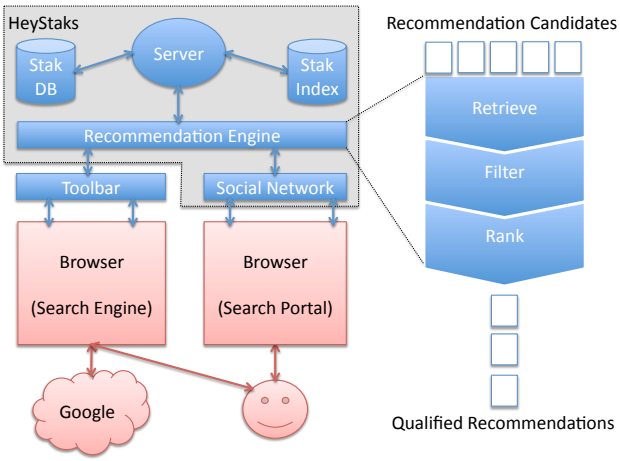


Fig. 2. The HeyStaks system architecture and outline recommendation model.

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 review how HeyStaks captures search activities within search staks and how this search knowledge is used to generate and filter result recommendations at search time; more detailed technical details can be found in [Smyth et al. 2009a; 2009b].

3.1 Profiling Stak Pages

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 ($S = \{r_1, ..., r_k\}$) and each result is anonymously associated with a number of implicit and explicit interest indicators, based on the type of actions that users can perform on these pages. A number of primary actions are facilitated, for example:

—*Selections (or Click-thrus)* – that is, a user selects a search result (whether *organic* or *recommended*). Similarly, HeyStaks allows a user to *preview* a page by opening it in a frame (rather than a window), and *popout* a page from a preview frame into a browser window;

—*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.

Result selections are an example of an *implicit* action in the sense that this type of action is part and parcel of normal routine search activity. It is also a weak indicator of relevance because users will frequently select pages that turn out to be

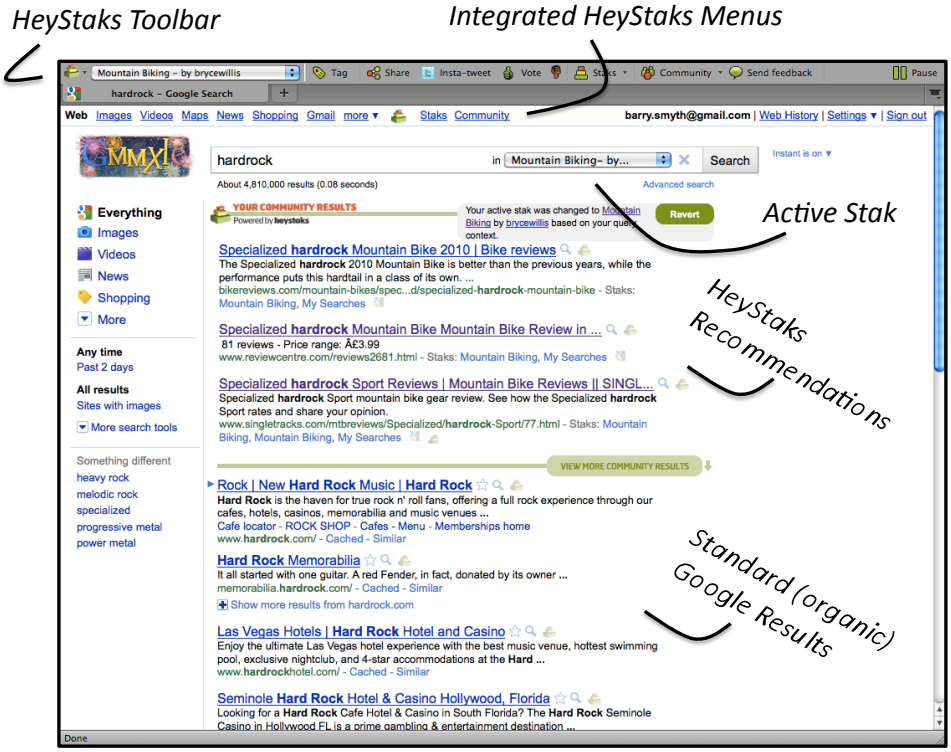


Fig. 3. HeyStaks in action: the screenshot shows how HeyStaks integrates seamlessly with mainstream search engines (Google in this case). In the example the searcher, a mountain biker, is looking for information from the specialist mountain biking brand, Hard Rock. The query submitted is clearly ambiguous and Google responds with results related to the restaurant/hotel chain. However, HeyStaks recognises the query as relevant to the *Mountain Biking* search stak that the searcher has previously joined and presents a set of more relevant results drawn from this stak.

irrelevant to their current needs. Nevertheless, the frequent selection of a specific page in a specific stak, in response to a particular type of query, suggests relevance. The 3 other forms of actions (voting, sharing, tagging) we refer to as *explicit* actions in the sense that they are not part of the normal search process, but rather they are HeyStaks specific actions that the user must chose to use. This type of deliberation suggests a stronger indicator of relevance and as such these actions are considered to be more reliable that simple result selections when it comes to evaluating the relevance of a page at recommendation time. Each result page r_i^S from stak S then, is associated with these indicators of relevance, including the total number of times a result has been selected (sel), the query terms $(q_1, ..., q_n)$ that led to its selection, the number of times a result has been tagged (tag), the terms used to tag it $(t_1, ..., t_m)$, the votes it has received (v^+, v^-), and the number of people it has been shared with ($share$) as indicated by Equation 1. This idea is related to earlier work by Amitay et al. [2005] and Smyth et al. [2004] which involve storing pages indexed by query terms. However, the present technology extends this to include

other indicators such as snippets, tags and votes.

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

In this way, each result 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 result indexed under its associated query and tag terms, and provides the basis for retrieving and ranking *recommendation 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 recommendation 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*.

3.2 Retrieval & Ranking

Briefly, there are two types of recommendation candidates: *primary recommendations* are results that come from the active stak S_t; whereas *secondary recommendations* come from other staks in the searcher's stak-list. To generate these recommendation 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 results $R(S_i, q_t)$. Each candidate result, r, is assigned a *relevance score* using a *TF*IDF*-based retrieval function as per Equation 2, which serves as the basis for an initial recommendation ranking.

$$relscore(q_t, r) = \sum_{t \epsilon q_t} tf(t \epsilon r) \times idf(t)^2 \ . \quad (2)$$

Staks are inevitably noisy, in the sense that they will frequently contain results that are not on topic. Thus, the retrieval and ranking stage may select results 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. 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; further detail can be found in [Smyth et al. 2009a; 2009b].

3.3 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 the next section we introduce our user reputation model, which is based on the collaboration events that inherently occur between users who share their search experiences. In turn, we show how this model can be employed to further enhance the quality of recommendations provided by HeyStaks by using reputation to influence the ranking of recommended results.

4. A REPUTATION MODEL FOR SOCIAL SEARCH

The many and varied different types of activities that a user can perform on a web page (click-thrus, tagging, voting, sharing) are ultimately combined and leveraged by HeyStaks to make recommendations at search time. And, while the recommendation algorithm described in Section 3 differentially weights different activity types (so that tagging, for example, is considered a more reliably indicator of interest that 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 than others and, as such, perhaps their activities should be considered more reliable at recommendation time. Thus recommendation 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. For example, as it stands it is feasible for a malicious user to flood a stak with results in the hope that at least some will be recommended to other users at search time. If unchecked this type of gaming has the potential to significantly degrade recommendation quality; see also recent related research on malicious users and robustness by the recommender systems community [Bryan et al. 2008; Lam and Riedl 2004; Mobasher et al. 2007; O'Mahony et al. 2002].

In the following section, 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. In turn we will describe how this reputation information can be combined with relevance to produce an improved recommendation engine, one that is capable of recommending results on the basis of their relevance to the user's query and stak context and according to the reputation of those users who were the source of these results within the staks in question.

4.1 From Activities to Reputation

It seems natural that the reputation of searchers 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 as a result of user activity. Each activity on the part of users causes 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 (perhaps by a different stak member), then the page's stak record will be updated to reflect the additional activity.

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 [Preece and Shneiderman 2009; Rashid et al. 2006]; related to this is the concept of trust in recommender systems and social networks [Kuter and Golbeck 2010; O'Donovan 2009] 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 consider search knowledge quality.

4.2 Reputation as Collaboration

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. Thus, our model of reputation must recognise the *quality* of *shared* search knowledge. There is a way to capture this notion of shared search by quality 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 this result (i.e. select, tag, vote or share), 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 simplest case, the original searcher, whose earlier action on this result caused it to be added to the search stak, and ultimately recommended, is known as the *producer*. In other words, the producer created search knowledge that was deemed to be relevant enough to be recommended and useful enough for the consumer to act upon it. 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 4(a)). 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 accumulate 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 events; 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 of high quality but also to share it with others.

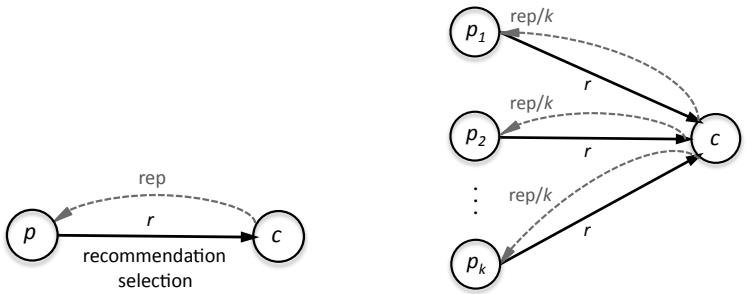


Fig. 4. Collaboration and Reputation: (a) the consumer c selects result r, which has been recommended based on the producer p's previous activity, so that c confers some unit of reputation (*rep*) on p. (b) More generally, the consumer c selects a result r that has been produced by a number of producers, $p_1, ..., p_k$, and reputation is shared amongst these producers with each user receiving an equal share of *rep/k* units of reputation.

4.3 A Computational Model of Reputation

The conferral of reputation by a single consumer on a single producer (Figure 4(a)) is the simplest case of our reputation model. More generally, at the time when the consumer acts (selects, tags, votes etc.) on the promoted result, there may have been a number of past producers who each contributed part of the search knowledge that caused this result to be promoted. A specific producer may have been the first to select the result in a given stak, 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. Thus we need to be able to share reputation across these different producers; see Figure 4(b).

More formally, let us consider the selection of a result r by a user c, the consumer, at time t. The producers responsible for the recommendation of this result are given by $producers(r, t)$ as per Equation 3 such that each p_i denotes a specific user in a specific stak.

$$producers(r, t) = \{p_1, ..., p_k\} \ . \tag{3}$$

Then, for each producer of r, p_i, we update its reputation as in Equation 4. In this way reputation is shared equally among its k contributing producers; see Figure 5 for an example of how user reputation can evolve over time.

$$rep(p_i, t) = rep(p_i, t-1) + 1/k \ . \tag{4}$$

Bear in mind that we are modeling user reputation at the *stak level*. Each user will have a separate reputation score for each stak in which they collaborate. When a result is recommended to a consumer it may originate from a number of different staks and so its producers may be members of different staks. Indeed the same user may be a producer of this result in more than one contributing stak. The above model ensures that user reputation scores are updated, at consumption time, on a stak by stak basis, thus ensuring that producers get credited based on their

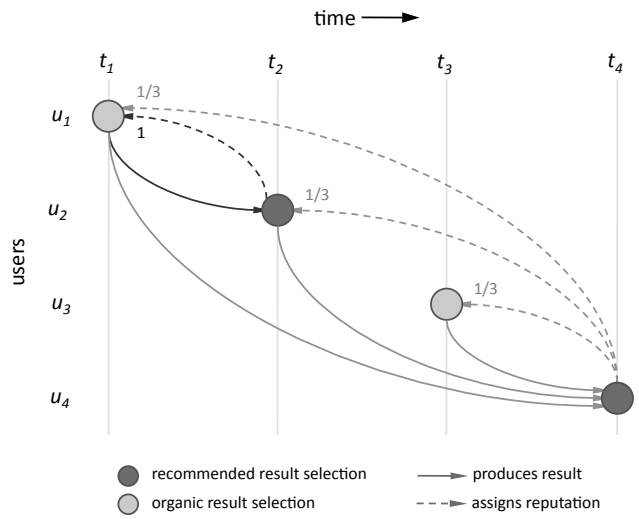


Fig. 5. The evolution of user reputation across users $u_1, ..., u_4$ for result page r, according to the reputation sharing strategy given by Equation 4. At time t_1, u_1 selects r causing it to be added to the stak. At time t_2, u_1 gains a single unit of reputation from u_2's selection of the recommended result r. At time t_3, r is independently added to the stak by the actions of u_3. Finally, at time t_4, r is again recommended and selected, this time by u_4, causing reputation to be shared equally between u_1, u_2 and u_3, resulting in u_1 having a final reputation score of 4/3 (1+1/3), u_2 and u_3 both having a score of 1/3 and u_4 having a score of 0.

stak contributions. This is important because it allows us to distinguish between different reputation levels for the same user in different staks, thereby reflecting different degrees of expertise across different subject matter. For example, the same user might be an expert when it comes to *Italian cuisine*, and enjoy a high reputation level in this stak, but might have little experience or knowledge when it comes to their new found love of *motorcycle maintenance*.

In Section 4.4 we will describe how this type of reputation model can be combined with result relevance at recommendation time with a view to providing a measure of protection against malicious users. Given the formulation of the reputation model, some protection against malicious activity is inherently provided because users only benefit if their results are recommended *and* selected by other users. Thus, even if recommended, irrelevant results are unlikely to be selected by consumers and the reputation of the malicious producer will not benefit, so that over time, the contributions of malicious users are less likely to be recommended in the future.

The reputation model as it stands is, however, susceptible to gaming in the following manner. To increase their reputation, malicious users could attempt to flood a stak with pages in the hope that at least some are recommended and subsequently acted on by other users. If this happens, then these malicious producers will benefit from increased reputation, and further pages from these users may continue to be recommended. The problem is that the current reputation model distributes reputation equally among all producers. To address this we can adjust our reputation model by changing the way in which reputation is distributed. The basic idea is

that a producer should receive more reputation if many of their past contributions have been consumed by other users but the should receive less reputation if most of their contributions have not been consumed.

More formally, for a producer p_i, let $n_t(p_i, t-1)$ be the total number of distinct results that this user has added to the stak in question prior to time t; remember that p_i refers to a single user and a specific stak. Further, let $n_r(p_i, t-1)$ be the number of these results that have been subsequently recommended and consumed by other users. We define the *consumption ratio* according to Equation 5; κ is an initialization constant that is set to 0.01 in our experiments. Accordingly, if a producer has a high consumption ratio it means that many of their contributions have been consumed by other users, suggesting that the producer has consistently added useful content to the stak. In contrast, if a user has a low consumption ratio then it means that few of their contributions have proven to be useful to other users.

$$consumption\_ratio(p_i, t) = \kappa + \frac{n_r(p_i, t-1)}{n_t(p_i, t-1)} \; . \tag{5}$$

Thus, given the selection of a result r by a consumer c at time t: if $p_1, ..., p_k$ are the contributing producers, then we can use their consumption ratios as the basis for sharing reputation according to Equation 6.

$$rep(p_i, t) = rep(p_i, t-1) + \frac{consumption\_ratio(p_i, t)}{\sum_{\forall p \epsilon \{p_1, ..., p_k\}} consumption\_ratio(p, t)} \; . \tag{6}$$

In this way, users who have a history of contributing many irrelevant results to a stak (that is, users with low consumption ratios) will receive a small proportion of the reputation share compared to users who have a history of contributing many useful results.

4.4 Reputation and Result Recommendation

In the previous sections we have described a reputation model for *users*. Individual stak members accumulate reputation when results that they have added to staks are recommended and acted on by other users. We have described how reputation is distributed between multiple producers during these collaboration events. In this section we describe how this reputation information can be used to produce better recommendations at search time.

In fact there are at least two ways in which this reputation information can be used. For example, we can implement a *reputation threshold* so that only results which originate from users with some minimum reputation score can be considered as recommendation candidates. We will return to this simple reputation threshold in the evaluation section that follows, but for now we will focus on a complementary mechanism to allow reputation information to influence recommendations.

The recommendation engine described in Section 3 operates at the level of an individual result page and scores each recommendation candidate based on how relevant it is to the target query. If we are to allow reputation to influence recommendation ranking, as well as relevance, then we need to transform our user-based

reputation measure into a result-based reputation measure. How then can we compute the reputation of a result that have been recommended by a set of producers?

One option is to simply add the reputation scores of the producers. However, this favours results that have been produced by lots of producers, even if the reputation of these producers is low. Another option is to compute the average of the reputation scores of the producers. However, this tends to depress the reputation of results that have been produced by many low-reputation users even if some users have very high reputation scores. In our work we have found a third option to work best. The reputation of a result page r (at time t) is simply the maximum reputation of its associated producers; see Equation 7. Thus, as long as at least some of the producers are considered reputable then this result will receive a high reputation score, even if many of the producers have low reputation scores. These less reputable users might be novices with respect to their knowledge of the stak topic and so their low reputations are not so much of a concern in the face of highly reputable producers.

$$repscore(r,t) = \max_{\forall p_i \epsilon \{p_1,...,p_k\}} \big(rep(p_i,t)\big) \; . \tag{7}$$

Now we have two ways to evaluate the appropriateness of a page for recommendation — the *relevance* of the page as per Equation 2 and its *reputation* as per Equation 7 — and we can combine these two scores using a simple weighted sum according to Equation 8 to calculate the rank score of a result page r and its producers $p_1, ..., p_k$ at time t, with respect to query q_t. The weight w varies between 0 and 1 and can be used to adjust the influence of relevance and reputation. For example, if $w = 0$ then recommended pages are ranked according to their relevance to the target query only, whereas if $w = 1$ then they are ranked by their reputation scores only. In the following section we will evaluate the rankings produced over a range of values for w.

$$rankscore(r, q_t, p_1, ..., p_k, t) = w \times repscore(r,t) + (1-w) \times relscore(q_t, r) \; . \tag{8}$$

5. EVALUATION

In this section we describe the results of a closed, live-user trial of HeyStaks, designed to evaluate the utility of HeyStaks' brand of collaborative search in fact-finding, information discovery tasks. In addition we also have the opportunity to evaluate the potential benefits of our new reputation model when it comes to boosting the relevance of HeyStaks' default promotions. It is worth highlighting that this present evaluation complements earlier evaluations of HeyStaks such as that carried out by Smyth et al. [2009b]. These earlier evaluations had the benefit of being open-ended trials, following users during routine search tasks, but were limited in their ability to evaluate the relevance of HeyStaks recommendations. Instead, these earlier evaluations reported on typical usage by HeyStaks users, focusing on stak creation and sharing behaviour. The benefit of the present closed trial is that it facilitates a more detailed comparative evaluation of result relevance, comparing HeyStaks recommendations to the default Google results, via a manual categorisation of the relevance of all the results acted on by all users during the course of the

No.	*Question*	*Answer*
1	Who was the last Briton to win the men's singles at Wimbledon	Fred Perry
2	Which Old Testament book is about the sufferings of one man	Job
3	Which reporter fronted the film footage that sparked off Band Aid	Michael Buerk
4	Which space probes failed to find life on Mars?	All of them
5	in the general theory of relativity what causes space-time to be modified?	Mass/Matter/Energy
6	Besides Hadrian, which Roman emperor had a wall built across Britain?	Antonine
7	Which person with "Strictly Come Dancing" links was involved in adapting "Saturday Night Fever" for stage?	Arlene Philips
8	The 18p stamp - the cheapest in the 1992 set - showed which Gilbert and Sullivan opera?	Yeomen of the Guard
9	Which Wimbledon winner was born the day Castro took over in Cuba?	John McEnroe
10	What is Daniel Defoe's real name?	Daniel Foe
11	Which town did Sky use as its UK satellite/cable testing ground?	Swindon
12	Javine Hylton, the UK's 2005 Eurovision entrant, once starred in which west end musical?	The Lion King
13	Who was the first British king to award medals to his troops for bravery?	Charles I
14	How many times was David Beckham sent off when playing for Man Utd?	Once
15	Which country has had more monarchs - Norway or Sweden?	Sweden
16	Who was the first artist to release a single with Madonna?	Britney Spears
17	What is the Australian name for a kind of long narrow lake?	Billabong
18	Which major UK sporting event took place the same day as Charles and Camilla's wedding?	The Grand National
19	The world's second TV service was beamed from which landmark?	Eiffel Tower
20	Which US secretary of defense held the post in two separate centuries?	Donald Rumsfeld

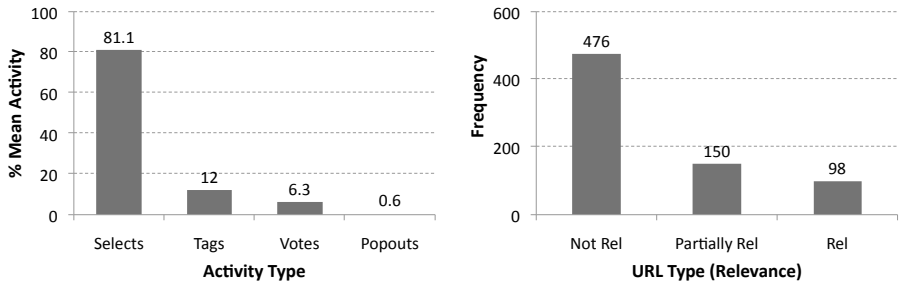
Table I. The questions presented to trial participants. The correct answers for each question are also shown.

trial.

5.1 Dataset and Methodology

Our experiment involves 64 first-year undergraduate university students with varying degrees of search expertise. Users were asked to participate in a general knowledge quiz, during a supervised laboratory session, answering as many questions as they could from a set of 20 questions in the space of 1 hour. The students worked concurrently on the same set of questions, which were randomly ordered to avoid any learning bias. The questions were selected from a quiz book by Preston and Preston [2007], and were chosen specifically for their obscurity and difficulty, and lead users to perform queries that are informational in nature. The questions and their correct answers are shown in Table I.

It was highly unlikely that students would be able to answer any significant number of these questions from their own general knowledge and so the purpose of this experiment was to look at how the students used HeyStaks and Google to help them answer these questions. Each user was allocated a desktop computer with Mozilla's Firefox web browser and the HeyStaks toolbar pre-installed; they were permitted to use Google, enhanced by HeyStaks functionality, as an aid in the quiz. Users were made aware of the functionality provided by the HeyStaks toolbar, so if they found a page they liked they could either *tag* it or *vote* on it, having been informed in an introductory one hour lecture and demonstration of the HeyStaks system how this might affect future Google searches and the searches of others. Note however that users were not explicitly directed to use the HeyStaks toolbar, rather to avail of it as they saw fit.



(a) (b)

Fig. 6. Summary statistics for the one hour user trial: (a) The mean percentage activity type per stak. (b) The number of irrelevant, partially relevant, and relevant pages found during the trial.

The 64 students were randomly divided into search groups. Each group was associated with a newly created search stak, which would act as a repository for the groups' search knowledge. We created 6 *solitary* staks, each containing just a single user, and 4 *shared* staks containing 5, 9, 19, and 25 users. The solitary staks served as a straightforward benchmark to evaluate the search effectiveness of individual users on a non-collaborative search setting, whereas the different sizes of shared staks provided an opportunity to examine the effectiveness of collaborative search across a range of different group sizes.

All activity on both Google search results and HeyStaks recommendations was logged, as well as all queries submitted during the experiment. Specifically, the following event/activity information was logged during the trial for later analysis:

—The time in which the activity took place (a Unix timestamp);

—The ID of the user who acted on a result and the stak ID in which the action was taken;

—The URL of the result page acted on;

—The type of action (result selection, tag, vote or share) performed;

—The type of result acted on, i.e. either an organic Google result or a HeyStaks recommended result.

During the 60 minute trial a total of 3,124 queries and 1,998 result activities (selections, tagging, voting, popouts) were logged, and 724 unique results were selected. As expected, during the course of the trial, result selections — the typical form of search activity — dominated over HeyStaks-specific activities such as tagging and voting. As shown in Figure 6(a), averaged across all staks, result selections accounted for just over 81% of all activities, with tagging accounting for just under 12% and voting for only 6%.

For the purpose of establishing a ground-truth for result relevance, each result page was examined post-trial by a number of experts and its relevance with respect to the appropriate quiz question was categorised as follows:

—not relevant (i.e. the result page content had no relevance with respect to a question);

—partially relevant (i.e. the result page contains an implicit reference to the answer or to a part of the answer to a question);

—relevant (i.e. the result page contains an answer to a question).

Figure 6(b) shows a relevance breakdown of the result pages logged during the course of the trial. 66% of result pages acted on were categorised as being not relevant with respect to the questions posed, while only 14% were deemed relevant. These findings demonstrate the difficulty of the questions presented as mentioned above. We will return to this relevance information later in this section when we use it to evaluate the relevance of HeyStaks recommendations.

5.2 Research Questions

Using this trial data we can explore a number of important questions pertaining to the benefits, or otherwise, of social web search and the value of reputation during result recommendation. In particular, in the remaining sections we will explore the following questions:

—*Is there evidence that search collaboration helps individual searchers find more relevant results than they might have on their own, in the absence of collaboration?* To answer this question we can look at the outcome of our quiz as the core search task. Overall, do students from shared search staks perform better than students from solitary search staks? Do the former attempt more questions than the latter? Do they answer more questions correctly?

—*How does collaboration influence the efficiency of search sessions?* For example, are there any differences in terms of the number of queries submitted or results selected (or tagged etc.) between solitary searchers and the collaborating searchers who are members of shared staks?

—*How good are the recommendations made by HeyStaks?* Specifically, how often can users expect to benefit from recommendations and, when recommendations are made, how relevant are they relative to the default organic results from the underlying search engine?

—*Does our reputation model offer a useful perspective on searcher reputation and/or expertise?* How do searchers in the trial accumulate reputation across shared staks? Do we see evidence of search leaders and followers? To what extent could this reputation model help to improve recommendation quality?

We will attempt to answer each of these questions with reference to the data from our live-user trial.

5.3 Quiz Performance

To begin with it is worth looking at the overall performance of students during the quiz as a basic outcome measure for this search task. Will the students participating in shared staks benefit from the searches of other stak members and outperform solitary searchers? And to what extent does stak size and the number of collaborators influence performance?

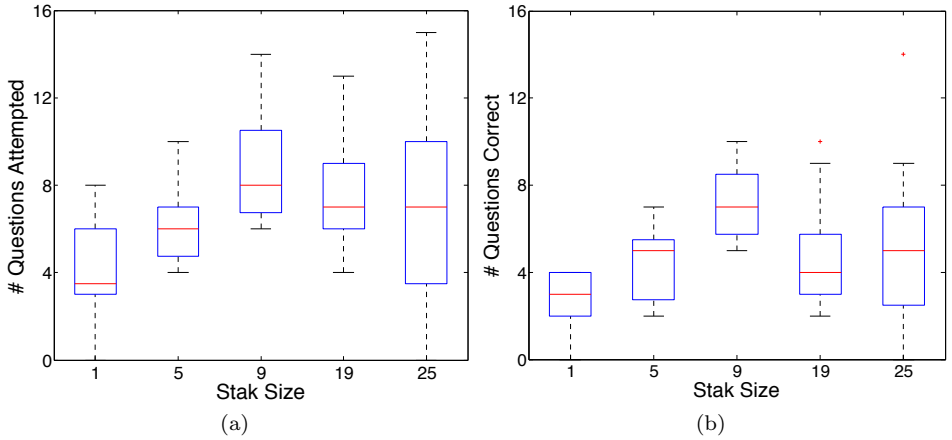


Fig. 7. Quiz performance: (a) The number of questions attempted per user per stak. (b) The number of questions correctly answered per user per stak.

Figure 7(a & b) presents box-plots of the number of questions attempted and answered correctly *per user* across the different stak sizes; note that for clarity we have grouped the results obtained for the 6 solitary staks and reported the aggregate information as a single solitary stak, indicated as the stak of size 1. These results point to benefit of the sharing and collaboration during this search task. For example, we see that the single-users of the 6 solitary staks attempt a median of 3.5 questions and answer only 3.0 of these questions correctly. By comparison, the median values across shared staks are between 5.5 and 8 questions attempted and between 4 to 7 questions correctly answered.

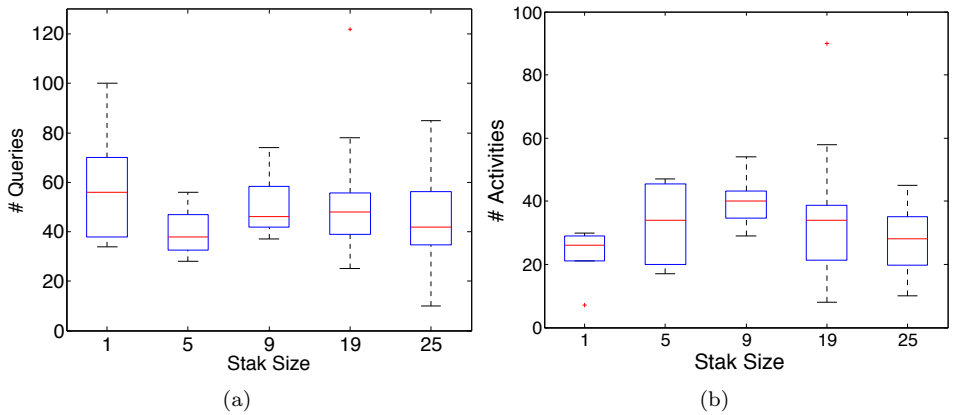
In general the influence of stak size is less clear in terms of these measures of overall performance. In the 9-person stak, more questions are answered correctly (7) than any of the other shared staks, for example, even compared to much larger 19-person and 25-person staks. It is likely that the search expertise of individual users is playing a role here and as such a simple measure such as stak size is unlikely to be a powerful predictor of overall performance given the variation in expertise that likely exists between between the individual members of a stak. Moreover, the closed-world nature of this trial — staks are limited by people and by topic to a 20-question quiz — likely limits the value of increasingly large staks, at least beyond some minimal critical mass.

5.4 Search Queries & Result Activities

We have presented evidence above to show how the members of our shared staks perform better than solitary searchers in our search task. Our key hypothesis is that this is due, at least in part, to the benefits of the type of search collaboration that HeyStaks is designed to facilitate. Specifically, we posit that the members of shared staks will benefit from relevant results, promoted due to the activities of other stak members, results that might otherwise be difficult to find. We will look in more detail at these promotions in the next section but first it is useful to look at the level of granular search activity across the different search staks. Are there

any differences between the numbers of queries submitted, or activities performed, by users across different stak sizes, for example?

Figure 8(a & b) presents box-plots for the number of queries and activities per user across the different search stak sizes; remember that by 'activity' we mean instances of users selecting, tagging, sharing or voting for results, as an indicator of relevance. We can view the number of queries submitted by a searcher as a proxy for their search effort and the number of activities (result selections, tagging, etc.) they generate to be an indicator of relevance for the results returned for these queries. To begin with we can see that the solitary searchers submit more queries than the students in the shared staks. Specifically, as per Figure 8(a), across these users the median number of queries submitted during the course of the search task is 52, compared with only 39 – 46 queries for users in the shared staks; or to put it another way, solitary searchers submit 13% – 33% more queries than their collaborating counterparts in shared staks. And when we look at the number of activities registered by solitary and collaborating users (Figure 8(b)) — as preliminary indicators of result relevance — we see the former have a median 23 activities (selections, tags etc.) across these queries compared to 28 – 40 activites for the members of the shared staks; this is a relative increase of 22% – 74% in favour of the shared staks.

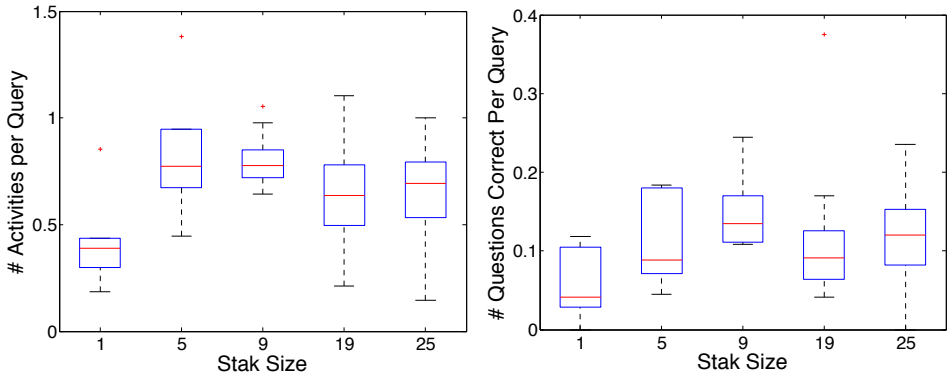


(a) (b)

Fig. 8. User query and activity performance: (a) The number of queries per user per stak. (b) The number of activities per user per stak.

An even clearer picture takes shape when we combine these two results to look at the median number of activities per query per user across the staks, as per Figure 9(a); this can be viewed as a proxy for the relevance (via number of activities) per unit search effort (number of queries submitted). Now we can see a very significant difference between the activities per query for the solitary searchers (approximately 0.4 activities per query) and the collaborating searchers in the shared staks (approximately 0.6 – 0.8 activities per query). In other words, 1.5 to 2-times as many queries lead to some form of activity among the users in shared staks compared to the solitary searchers, suggesting that the former are benefitting

significantly from results that are, apparently at least, more relevant than those experienced by the latter. Again we will return to the notion of relevance in a future section but for now perhaps an even more pragmatic metric of relevance per unit search effort can be calculated by combining the average number of correct quiz answers per query per user across the various staks. This is presented in Figure 9(b) and once again we can see a very significant difference between the solitary searchers and the users who are members of shared staks. In the case of the former, on average they correctly answer 0.044 questions per query, but for the latter this ratio increases to 0.15. In other words, on a per query basis our collaborating searchers are answering up to more than 3 times as many questions correctly than the solitary searchers, which is a very significant productivity-gain for the members of shared staks.



(a) (b)

Fig. 9. Two measures of relevance per unit search effort across staks: (a) The number of activities per query per user. (b) The number of correct answers per query per user.

5.5 Recommendations & Relevance

Given that the members of shared search staks seem to be enjoying improved search productivity when compared to their solitary counterparts, we now turn our attention to likely source of this improvement: the recommendations that are generated by HeyStaks. To begin with it is worth looking at how often HeyStaks is able to recommend results to the members of the different staks. This is presented in Figure 10 as the percentage of queries that result in at least one HeyStaks recommendation. As expected, larger staks mean more recommendations, because there are more search experiences to act as a source of recommendations. For example, for the solitary staks, we find that only 16% of the queries lead to recommendations, and, by definition, these recommendations are due to the solitary searcher submitting queries that are similar to those they have used previously. In contrast, the likelihood of recommendations grows quickly as stak size increases. Even for

the 5-person stak, nearly 40% of queries lead to recommendations, growing to over 62% for the largest 25-person stak.

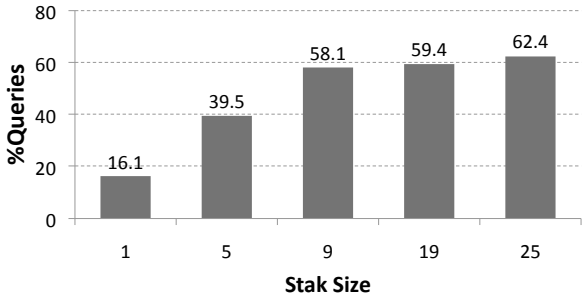


Fig. 10. Number of queries receiving at least one HeyStaks recommendation as a percentage of total queries submitted per stak.

Of course simply making lots of recommendations is not the goal of HeyStaks. The success of these recommendations will depend on how relevant they are and, in particular, whether they are more relevant than the default, organic results from Google. To explore this we focus on those search results that ultimately received user attention (selections, tags etc). There are 724 of these results and, as mentioned previously, we manually categorised each as *relevant*, *partially relevant*, or *not relevant*. Figure 11(a & b) shows the percentage of these result activities that are relevant, partially relevant, and not relevant for both the default (Google) organic results and the HeyStaks recommendations across the shared staks. In this case we exclude single-person staks as we wish to examine the effects of result-sharing, rather than simply result recovery.

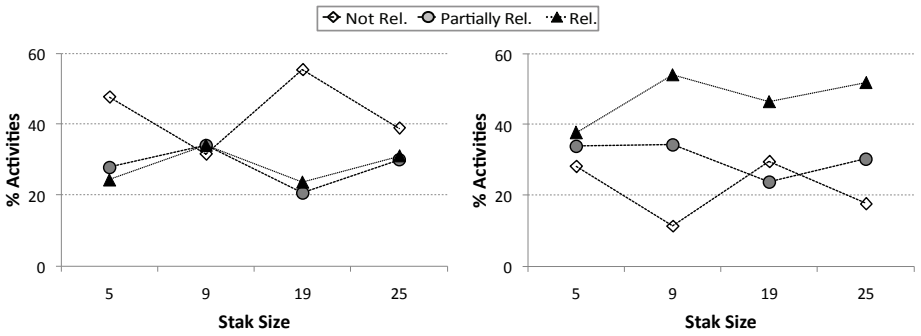


Fig. 11. The relevance of (a) organic and (b) recommended results acted on per stak.

Comparing the graphs for the recommended results versus the organic results we can see a significant relevance benefit for the former. For example, an average of

48% of recommended result activities (averaged across the 4 different stak sizes) are deemed to be relevant compared to only 28% for the organic results; in other words, the recommendations that attract user activity tend to be more frequently relevant than the organic results that attract user activity. Similarly, we find that, on average, 41% of the organic result activities are for not relevant results compared to only 21% for the recommended result activities.

$$relevance\ ratio = \frac{a_r}{a_{nr}} \tag{9}$$

To better quantify this relevance benefit we can compute a *relevance ratio* for organic and recommended results as per Equation 9. Basically, this is the ratio of relevant results to not relevant results. A relevance ratio less than 1 means that the majority of results are not relevant, whereas a relevance ratio of more than 1 means that the majority of results are relevant. Figure 12 presents the relevance ratio of organic and recommended results on a stak by stak basis. For each stak we can see that the recommended results have a much higher relevance ratio than the default organic results. For example, in the case of the 5-person stak, the organic results have a relevance ratio of 0.5. However, the relevance ratio for the recommended results in this stak is more than twice as high, at 1.3.

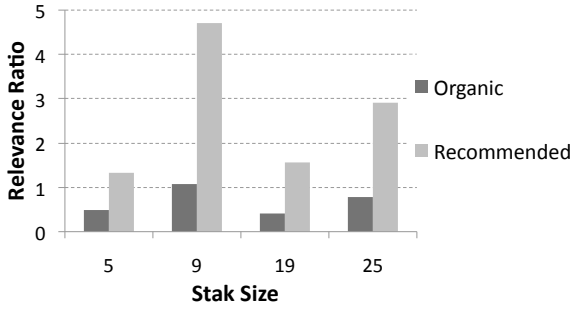


Fig. 12. The relevance ratio for organic and recommended results per stak.

It is worth discussing the 9-person stak, which does especially well by this evaluation measure. Although not the largest stak, this stak is the best performer (e.g. more questions answered correctly per user), most likely because its members are better searchers to begin with. The relevance ratio of its recommended results is 4.7 (about 55% of these results are relevant) which points to the very high quality of the recommendations made for this group of searchers. But it is interesting to note that for this stak the relevance ratio of the organic results is also relatively high, at 1.1. This supports the notion that these stak members are better than the average searchers. Even their organic search results are more relevant than the norm, presumably because they are able to produce more effective queries in the first instance. And of course if the resulting organic results are more relevant to begin with, then this will ultimately translate into superior recommended results

because these more-relevant organic results ultimately become recommendations themselves as they are acted on by users.

5.6 Searcher Reputation

The results of the previous section highlight the potential benefits of the HeyStaks form of collaborative web search in the context of the target search task. Recommended results turned out to be significantly more relevant, according to our independent relevance metric, than conventional organic results. In effect we found Heystaks to be amplifying the relevance of organic results through its recommendation process, with better quality organic results leading to a progressive uplift in the quality of the results that make it through the various recommendation stages and filters. As an aside, the correlation between the organic relevance ratio and the recommendation relevance ratio data from Figure 12 is 0.98, indicating a strong linear relationship between the quality of the organic results and the subsequent quality of the recommended results. This bodes well for HeyStaks as it means that recommendation relevance is fuelled by search expertise within a stak, which creates a kind of positive feedback loop in the drive towards better recommendations. However, this type of positive feedback is not without its dangers and one obvious problem is that, on its own, it could provide a mechanism for malicious users to spam a stak and accelerate the promotion of their target content. Even absent overtly malicious users, recommendation quality can degrade if prolific, but inexperienced, searchers contribute large quantities of irrelevant results to a stak.

Clearly there needs to be some sort of control to protect against such issues in practice and it is with this in mind that we have developed the reputation model described earlier in Section 4, which provides a mechanism to differentiate between recommendations that are derived from the activities of inexperienced versus experienced or malicious versus well-meaning users. In this section we explore the reputation data generated during the live-user trial. In what follows, reputation is shared among the producers in a collaboration event in proportion to the quality of their previous search contributions as described in Section 4.3. We will focus our reputation analysis on the 58 users who were members of the shared staks, since by definition the reputation model does not apply to searchers in solitary staks.

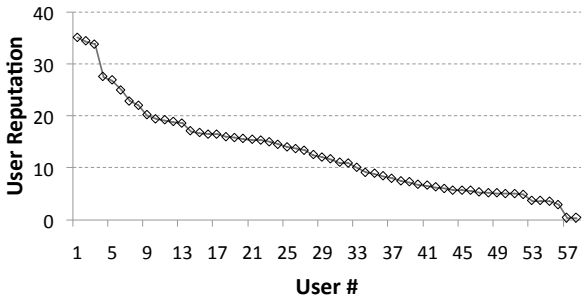


Fig. 13. Reputation scores for the 58 users participating in shared staks.

Figure 13 plots the reputation scores, accumulated by the end of the trial, across the 58 collaborating searchers. Recall that the reputation score of a user is effectively a function of the frequency with which their contributions have been recommended and subsequently selected (or otherwise acted on) by other searchers. Clearly there is a diverse range of reputation scores across all of these users. All users have a reputation score greater than zero, indicating that everyone contributed to at least some collaboration events within their respective staks. Interestingly, there are a number of users with especially high reputation scores: The top 8 users have reputation scores of 20 or more, indicating that they acted as producers for at least 20 collaboration events, and likely many more depending on how many other producers were also involved in the same events, which, as described in Section 4.3, affects the distribution of reputation across producers. In fact, the mean number of producers per collaboration event is 3.4, with a standard deviation of 2.3.

We might ask where these reputation scores come from for our users. For example, do they accrue from just a small number of key results that many other users select when they are recommended? Or do we find that different users are broadly contributing to search expertise across a wider variety of results? As it turns out, the latter is the case. Figure 14 plots the reputation score of a user versus the number of distinct results contributed to collaboration events by that user. We can see that the more reputable users contribute more distinct results (from 10 to 27 distinct results), which then serve as valuable search knowledge to drive effective recommendation and collaboration.

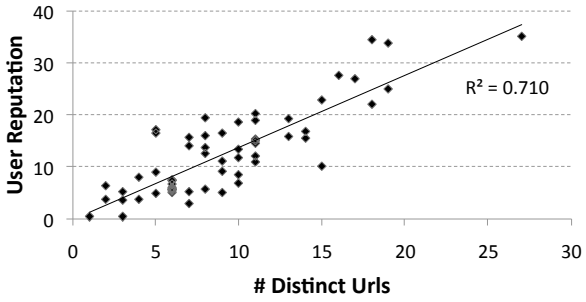


Fig. 14. Reputation score vs. distinct number of results produced.

Turning our attention to reputation at the stak-level, Figure 15 presents box-plots for the reputation scores per user across the 4 shared staks. As we might expect we can immediately see how larger staks tend towards higher median reputation scores across their members — more members means more opportunity for collaboration and thus higher reputation potential — but this tendency does not always hold. For example, we can see that the median reputation score for members of the 5-person stak is approximately 5 compared to an average median reputation score of about 13 for the larger 9, 19 and 25-person staks. The most reputable user in the trial, with a reputation score of 35, hails from the 19-person stak; however, the next two most reputable users, both with reputation scores in excess of 30, are

members of the 9-person stak, which has a very similar median reputation score (14.5) despite having ten fewer members compared to the 19-person stak (14.9). We know from our earlier performance results that the users in the 9-person stak perform particularly well, both in terms of their quiz performance (e.g. median questions correct per queries submitted) and the relevance of their search results. This performance is reflected in their reputation scores too. Moreover, the box-plot for the 9-person stak indicates a higher reputation-score at the first and third quartiles than is found for any of the other staks.

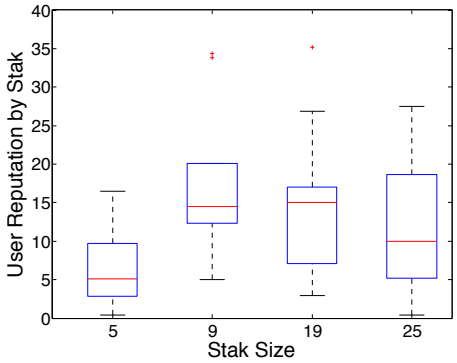


Fig. 15. User reputation score per user per (shared) stak.

The above data relate to the reputation scores that accumulated by the end of the 60-minute trial. It is also interesting to look at how reputation builds during the course of the trial. For example, is there a slow accumulation of reputation, indicating that effective collaboration is rare on the ground during the early stages of the trial? Or does effective collaboration start from an early stage, in which case we should find a more rapid growth in reputation among stak members. To examine this we note the number of users with non-zero reputation score at 5-minute intervals during the trial; we do this retrospectively by analysing the collaboration logs. The results are plotted in Figure 16 for each of the 4 shared staks across the 60 minute duration of the trial (from 10.30 am to 11.30 am). We see a consistent reputation profile across the 4 staks with reputation beginning to accumulate from an early stage, albeit more slowly, as expected, for the 5-person stak. For example, by the 20-minute mark, the 9, 19, and 25-person staks all have in excess of 80% of their members with non-zero reputation, compared to 40% for the 5-person stak. And for these 3 larger staks, 100% of their members have non-zero reputation by about half-way through the trial. In other words, most stak members contribute useful search knowledge to staks from a very early stage so that other members start to benefit from useful recommendations from very early on during the trial. It is interesting to note that the 9-person stak again performs particularly well according to this measure, and indeed it outperforms the largest 25-person stak. Hence these findings provide an indication that stak size is not the sole determining factor when it comes to collaboration between users; the quality of search content is also likely to play a key role in this regard.

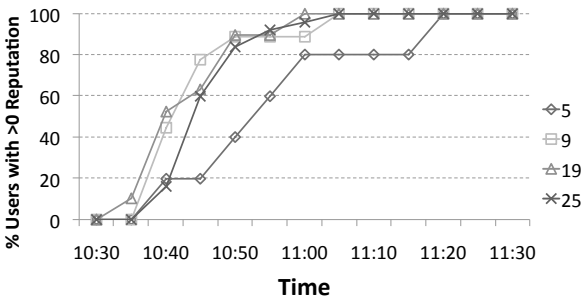


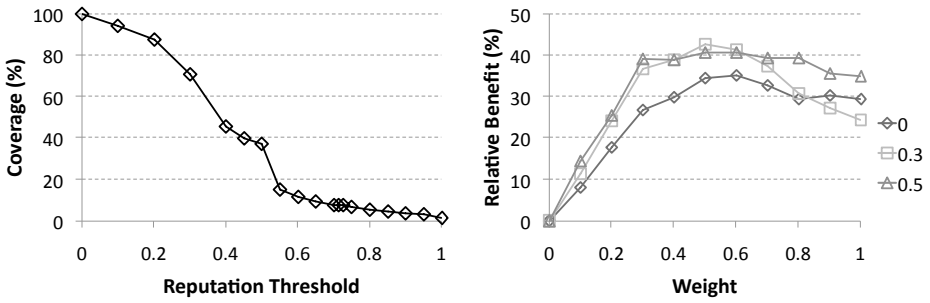
Fig. 16. Percentage of users with >0 reputation score per stak vs. time.

5.7 Reputation for Recommendation Ranking

As discussed previously the motivation for incorporating a reputation model into the HeyStaks recommendation engine is to provide a way for searcher expertise to influence recommendation. In Section 4.4 we described how to calculate the reputation score of a result page from its producers and how to incorporate this into the recommendation ranking process, by combining result reputation with the default HeyStaks' relevance score. During the present trial we did not include the reputation model directly for the purpose of ranking recommendations; the results presented up until now are all based on the default Heystaks relevance-based recommendation system described in Section 3.2. However, because we can compute the reputation of users at any point during the course of the trial, it is feasible to retrospectively apply the reputation model to re-rank the HeyStaks recommendations in order to assess the relevance of the re-ranked recommendations in comparison to the default ranking.

To do this we simply re-ranked the recommendations for every trial query using Equation 8 in Section 4.4. We adjusted the reputation weight, w, from 0 (no reputation) to 1 (pure reputation ranking) to examine the effect of modulating the influence of reputation compared to the default HeyStaks recommendation score. In addition we tested a *reputation filter* to eliminate any recommendations which had less than a pre-defined *reputation threshold*. In principle, by increasing the reputation threshold in this way we should experience an improvement in recommendation quality, but at the same time it will reduce recommendation coverage — the number of recommendations that can be made — because none of the recommendations for certain queries will exceed the threshold. The effect of this is presented in Figure 17(a) as a graph of coverage versus reputation threshold. It is clear that as the reputation threshold increases there is a steady decline in coverage. Obviously there is little to be gained from increasing the reputation threshold to such an extent that no, or very few, recommendations can be made and so for the purpose of this experiment we consider reputation thresholds of 0 (providing 100% coverage because all result pages have reputation ≥ 0), 0.3 (providing coverage of approximately 70%) and 0.5 (providing coverage of just under 40%).

For the purpose of a side-by-side comparison of the standard HeyStaks recommendation ranking versus the variations (by reputation weight and reputation thresh-



(a) (b)

Fig. 17. Reputation ranking: (a) The percentage of recommended results (coverage) with a reputation score exceeding a given reputation threshold. (b) The relative benefit (percentage increase in relevance ratio) across different reputation weights and thresholds.

old) on reputation-based ranking we calculated the relevance ratio across all top-ranked recommendations made by each system for all queries submitted during the user trial. Relevance ratio is calculated as the number of relevant pages recommended divided by the number of not relevant pages recommended as per Equation 9. The results of this experiment are presented in Figure 17(b) as the *relative benefit* (percentage increase in relevance ratio) of reputation-based ranking, in comparison to the default HeyStaks recommendation ranking, for different values of the reputation weight (w), from 0 to 1, and for 3 different reputation thresholds (0, 0.3, and 0.5). For example, according to Figure 17(b), we see that at a reputation threshold of 0 and a reputation weight of 0.5, there is a relative benefit of 35%. At these settings, the relevance ratio for default HeyStaks' recommendations was 1.25, and the relevance ratio for recommendations using reputation-based ranking was 1.69, leading to a relative benefit of $(1.69 - 1.25)/1.25$ or 35%.

Clearly the results for Figure 17(b) speak to the significant benefits that can be gained by integrating our reputation model with the default HeyStaks' recommendation engine. We can see that across all of the reputation weights evaluated, once reputation is allowed to influence the recommendation ranking (that is, once the reputation weight is greater than 0) then there is an increase in the relative number of top-ranked recommendations that are judged to be relevant. Based on the reputation data generated in this trial, the optimal reputation weighting seems to be in the 0.4 – 0.6 region (with relative benefits in the 35% – 45% range) allowing reputation to play a more or less equal role to the default HeyStaks' recommendation score during ranking. As the reputation weight is increased, initially we see a rapid increase in its relative benefit score but as the reputation weight exceeds 0.6 we see relative benefit fall back as it begins to over-influence the recommendation rankings. As expected there is a benefit due to increasing the reputation threshold: the relative benefit curves for the 0.3 and 0.5 thresholds both outperform the 0 threshold setting. However there is little real difference in the outcome between the 0.3 and 0.5 thresholds, at least in this experiment, most likely because of the limits

of the data available during this trial.

5.8 Limitations & Results Summary

In this evaluation we have described the results of a live-user trial of HeyStaks. Importantly we acknowledge that this trial is limited and that our results must be viewed in the context these limitations. It is not a large-scale trial of thousands or millions of searchers. Such a trial might be possible in the context of conventional search engines but it is not feasible, at least not yet, for HeyStaks. Nevertheless the trial does involve a reasonable number of users and reflects a realistic search use-case. Of course this use-case — a fact-finding search task — also has its limitations. It is, for example, just one of the many reasons why users avail of search engines and there is clearly an opportunity for further work in order to broaden our evaluation to cover more open-ended search and discovery tasks; preliminary results for these open-ended style evaluations have been presented elsewhere in Smyth et al. [2009b]. Nevertheless, our closed quiz does provide useful insight and facilitates a thorough evaluation with respect to an independent model of result relevance, and as such we could state definitively which results were relevant and which were not relevant, on a question-by-question basis.

Given these trial limitations, the outcome of our evaluation has been very positive. We have demonstrated that there are clear benefits for those users who participated in shared staks compared to solitary searchers. The former enjoyed improved search performance overall and required significantly less search effort. The evaluation helped to clarify the relevance benefits of HeyStaks recommendations. Shared stak members benefited from recommendations that were objectively more relevant than the default organic search results. These recommendations effectively amplified the relevance of results selected by search leaders and benefitted search followers accordingly.

Finally, we demonstrated the benefits of our proposed reputation model in a very concrete, albeit offline, manner: by allowing reputation to influence recommendation ranking it was possible to significantly improve the relevance of the top-ranked recommendations made to users. Of course we are not able to conclude that this will mean that searchers are likely to benefit directly from this improved ranking, because we were not in a position to evaluate the actual responses of live users to these re-ranked recommendations. It is conceivable, for example, that searchers may avoid these more relevant results when they are ranked using reputation, while selecting them in the default HeyStaks ranking. However, this seems most unlikely and it is common practice in web search evaluations to acknowledge that there is an extremely strong bias between the position of results and their likelihood of selection (see e.g. Keane et al. [2008]) and, as such, it is generally accepted that if one can produce rankings where top-ranked results are more relevant, then these rankings are likely to meet with a better user response. Hence we believe that the findings of the previous section have merit when considered from this viewpoint.

6. CONCLUSIONS

The world of web search is changing. Many of our information needs are being met by sharing through social networks as much as they are through queries to search engines. As web search evolves there is a significant opportunity for search engines

to accommodate a more collaborative form of information discovery, one that takes advantage of our social networks to deliver an improved search experience that can be influenced by our trusted friends and reputable third-parties.

To this end we have described the HeyStaks social search service. HeyStaks supports collaborative web search by allowing the past search experiences of our friends and colleagues to influence our future searches. It does this by providing a segmented social search experience in which individual users can create and share search staks on topics of their choosing. At search time, HeyStaks learns from the search activities of the members of a stak and uses this information to generate recommendations based on results that other users have recently found relevant for similar searches. HeyStaks delivers this social search functionality via the browser so that users can continue to use their favourite mainstream search engine while benefiting from a more collaborative search experience. The core contribution of this paper has been an extension of the HeyStaks recommendation engine which incorporates a novel model of search reputation, based on the extent to which a user contributes to collaboration across the staks of which they are members.

We have also described a live-user trial of HeyStaks to demonstrate the relevance of its core recommendations across different types of search stak, and the value of the reputation model as a way to further improve recommendation quality. Overall the results of this trial speak to the clear benefits of this more collaborative approach to web search. Collaborating searchers demonstrated improved performance in the benchmark task and an objective evaluation of result relevance indicates that the HeyStaks recommendations enjoyed superior relevance to the default Google results. Moreover, we demonstrated how the reputation model quickly helped to distinguish the most experienced searchers from those less experienced, and by incorporating reputation into the recommendation process it was possible to further improve the relevance of recommendations by over 40%.

Finally, it is worth highlighting that HeyStaks is a robust, scalable social search service that has been designed not as a laboratory testbed but rather as a deployable social search service. To this end the service is currently available in beta form at www.heystaks.com.

REFERENCES

Amershi, S. and Morris, M. R. 2008. Cosearch: a system for co-located collaborative web search. In *CHI*. 1647–1656.

Amitay, E., Darlow, A., Konopnicki, D., and Weiss, U. 2005. Queries as anchors: selection by association. In *In Proc. HYPERTEXT*. ACM Press, 193–201.

Asnicar, F. A. and Tasso, C. 1997. Ifweb: a prototype of user model-based intelligent agent for document filtering and navigation in the world wide web. In *Proceedings of the workshop "Adaptive Systems and User Modeling on the World Wide Web, Sixth International Conference on User Modeling"*. 3–11.

Boydell, O. and Smyth, B. 2010. Social summarization in collaborative web search. *Information Processing and Management*.

Bryan, K., O'Mahony, M., and Cunningham, P. 2008. Unsupervised retrieval of attack profiles in collaborative recommender systems. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*. ACM, New York, NY, USA, 155–162.

Budzik, J. and Hammond, K. J. 2000. User interactions with everyday applications as context for just-in-time information access. In *IUI '00: Proceedings of the 5th international conference on Intelligent user interfaces*. ACM, New York, NY, USA, 44–51.

Chang, H., Cohn, D., and McCallum, A. 2000. Learning to create customized authority lists. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 127–134.

Chirita, P. A., Nejdl, W., Paiu, R., and Kohlschütter, C. 2005. Using odp metadata to personalize search. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM Press, New York, NY, USA, 178–185.

Chirita, P.-A., Olmedilla, D., and Nejdl, W. 2004. Pros: A personalized ranking platform for web search. In *AH*, P. D. Bra and W. Nejdl, Eds. Lecture Notes in Computer Science, vol. 3137. Springer, 34–43.

Evans, B. M. and Chi, E. H. 2009. An elaborated model of social search. *Information Processing and Management In Press, Corrected Proof*, –.

Evans, B. M., Kairam, S., and Pirolli, P. 2010. Do your friends make you smarter?: An analysis of social strategies in online information seeking. *Information Processing and Management*.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. 2001. Placing search in context: the concept revisited. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*. ACM, New York, NY, USA, 406–414.

Golovchinsky, G., Qvarfordt, P., and Pickens, J. 2009. Collaborative information seeking. *IEEE Computer 42,* 3.

Granka, L. A., Joachims, T., and Gay, G. 2004. Eye-tracking analysis of user behavior in www search. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, 478–479.

Hoffman, K., Zage, D., and Nita-Rotaru, C. 2009. A survey of attack and defense techniques for reputation systems. *ACM Comput. Surv. 42,* 1.

Jøsang, A. and Golbeck, J. 2009. Challenges for robust trust and reputation systems. In *5th International Workshop on Security and Trust Management*.

Jøsang, A., Ismail, R., and Boyd, C. 2007. A survey of trust and reputation systems for online service provision. *Decis. Support Syst. 43,* 2, 618–644.

Keane, M. T., O'Brien, M., and Smyth, B. 2008. Are People Biased In Their Use Of Search-Engines? *Communications of the ACM 51,* 2, 49–52.

Kuter, U. and Golbeck, J. 2010. Using probabilistic confidence models for trust inference in wb-based social networks. *ACM Transactions on Internet Technologies 10,* 2, 1–23.

Lam, S. K. and Riedl, J. 2004. Shilling recommender systems for fun and profit. *In Proceedings of the 13th International World Wide Web Conference*, 393–402.

Lawrence, S. and Giles, C. L. 1998. Context and page analysis for improved web search. *IEEE Internet Computing 2,* 4, 38–46.

Lazzari, M. 2010. An experiment on the weakness of reputation algorithms used in professional social networks: The case of naymz. In *IADIS International Conference e-Society*. 519–522.

Ma, Z., Pant, G., and Sheng, O. R. L. 2007. Interest-based personalized search. *ACM Trans. Inf. Syst. 25,* 1, 5.

Makris, C., Panagis, Y., Sakkopoulos, E., and Tsakalidis, A. 2007. Category ranking for personalized search. *Data Knowl. Eng. 60,* 1, 109–125.

Massa, P. and Avesani, P. 2007. Trust-aware recommender systems. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*. ACM, 17–24.

Mobasher, B., Burke, R., Bhaumik, R., and Williams, C. 2007. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transactions on Internet Technology 7,* 4.

Morris, M., Teevan, J., and Panovich, K. 2010. What do people ask their social networks, and why? a survey study of status message q and a behavior. In *Computer Human Interaction*.

Morris, M. R. and Horvitz, E. 2007a. S^3: Storable, shareable search. In *INTERACT (1)*. 120–123.

Morris, M. R. and Horvitz, E. 2007b. Searchtogether: an interface for collaborative web search. In *UIST*. 3–12.

O'Donovan, J. 2009. Capturing trust in social web applications. *Computing with Social Trust*, 213–257.

O'Donovan, J. and Smyth, B. 2005. Trust in recommender systems. In *Intelligent User Interfaces*. 167–174.

O'Donovan, J. and Smyth, B. 2006. Is trust robust? an analysis of trust-based recommendation. In *Intelligent User Interfaces*. 101–108.

O'Mahony, M. P., Hurley, N. J., and Silvestre, G. C. M. 2002. Promoting recommendations: An attack on collaborative filtering. In *DEXA*. 494–503.

Pickens, J., Golovchinsky, G., Shah, C., Qvarfordt, P., and Back, M. 2008. Algorithmic mediation for collaborative exploratory search. In *SIGIR*. 315–322.

Preece, J. and Shneiderman, B. 2009. The reader to leader framework: Motivating technology-mediated social participation. *AIS Trans. on Human-Computer Interaction 1,* 1, 13–32.

Preston, R. and Preston, S. 2007. *The Official Biggest Pub Quiz Book Ever!* Carlton Books Ltd.

Pretschner, A. and Gauch, S. 1999. Ontology based personalized search. In *ICTAI '99: Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*. IEEE Computer Society, Washington, DC, USA, 391.

Rashid, A. M., Ling, K., Tassone, R. D., Resnick, P., Kraut, R., and Riedl, J. 2006. Motivating participation by displaying the value of contribution. In *Computer Human Interaction*. 955–958.

Resnick, P. and Zeckhauser, R. 2002. Trust among strangers in internet transactions: Empirical analysis of ebay's reputation system. *Advances in Applied Microeconomics 11*, 127–157.

Resnick, P., Zeckhauser, R., Friedman, E., and Kuwabara, K. 2000. Reputation systems: Facilitating trust in internet interactions. *Communications of the ACM 43,* 12, 45–48.

Sabater, J. and Sierra, C. 2005. Review on computational trust and reputation models. *Artificial Intelligence Review 24,* 1, 33–60.

Shen, X., Tan, B., and Zhai, C. 2005. Implicit User Modeling for Personalized Search. In *Proceedings of the Fourteenth ACM Conference on Information and Knowledge Management (CIKM 05)*.

Signorini, A. and Gulli, A. 2005. The indexable web is more than 11.5 billion pages. In *Special interest tracks and posters of the 14th international conference on World Wide Web*. 902–903.

Smeaton, A. F., Foley, C., Byrne, D., and Jones, G. J. F. 2008. ibingo mobile collaborative search. In *CIVR*. 547–548.

Smeaton, A. F., Lee, H., Foley, C., and McGivney, S. 2007. Collaborative video searching on a tabletop. *Multimedia Syst. 12,* 4-5, 375–391.

Smyth, B., Balfe, E., Boydell, O., Bradley, K., Briggs, P., Coyle, M., and Freyne, J. 2005. A live-user evaluation of collaborative web search. In *IJCAI*, L. P. Kaelbling and A. Saffiotti, Eds. Professional Book Center, 1419–1424.

Smyth, B., Balfe, E., Freyne, J., Briggs, P., Coyle, M., and Boydell, O. 2004. Exploiting query repetition and regularity in an adaptive community-based web search engine. In *UMUAI*. 383–423.

Smyth, B., Briggs, P., Coyle, M., and O'Mahony, M. P. 2009a. A case-based perspective on social web search. In *International Conference on Case-Based Reasoning*. 494–508.

Smyth, B., Briggs, P., Coyle, M., and O'Mahony, M. P. 2009b. Google? shared! a case-study in social search. In *User Modeling, Adaptation and Personalization*. Springer-Verlag.

Song, R., Luo, Z., Wen, J.-R., Yu, Y., and Hon, H.-W. 2007. Identifying ambiguous queries in web search. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*. ACM Press, New York, NY, USA, 1169–1170.

Speretta, M. and Gauch, S. 2005. Personalized search based on user search histories. In *WI '05: Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*. IEEE Computer Society, Washington, DC, USA, 622–628.

Spink, A. and Jansen, B. J. 2004. A study of web search trends. *Webology 1,* 2, 4.

Zhou, B., Hui, S. C., and Fong, A. C. M. 2006. An effective approach for periodic web personalization. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*. IEEE Computer Society, Washington, DC, USA, 284–292.