<table>
<thead>
<tr>
<th>Title</th>
<th>Evaluating user reputation in collaborative web search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authors(s)</td>
<td>McNally, Kevin; O'Mahony, Michael P.; Smyth, Barry</td>
</tr>
<tr>
<td>Publication date</td>
<td>2011-10-23</td>
</tr>
<tr>
<td>Publication information</td>
<td>Freyne, J. et al. (eds.). Proceedings of the 3rd ACM RecSys 10 Workshop on Recommender Systems and the Social Web</td>
</tr>
<tr>
<td>Link to online version</td>
<td><a href="http://www.dcs.warwick.ac.uk/~ssanand/RSWeb11/rsweb2011proceedingsfinal.pdf">http://www.dcs.warwick.ac.uk/~ssanand/RSWeb11/rsweb2011proceedingsfinal.pdf</a></td>
</tr>
<tr>
<td>Item record/more information</td>
<td><a href="http://hdl.handle.net/10197/3478">http://hdl.handle.net/10197/3478</a></td>
</tr>
</tbody>
</table>
Evaluating User Reputation in Collaborative Web Search

Kevin McNally, Michael P. O’Mahony, Barry Smyth
CLARITY Centre for Sensor Web Technologies
School Of Computer Science & Informatics
University College Dublin
{firstname.lastname}@ucd.ie

ABSTRACT
Often today's recommender systems look to past user activity in order to influence future recommendations. In the case of social web search, employing collaborative recommendation techniques allows for personalization of search results. If recommendations arise from past user activity, the expertise of those users driving the recommendation process can play an important role when it comes to ensuring recommendation quality. Hence the reputation of users is important in collaborative and social search tasks, in addition to result relevance as traditionally considered in web search. In this paper we explore this concept of reputation; specifically, investigating how reputation can enhance the recommendation engine at the core of the HeyStaks social search utility. We evaluate a number of different reputation models in the context of the HeyStaks system, and demonstrate how incorporating reputation into the recommendation process can enhance the relevance of results recommended by HeyStaks.

1. INTRODUCTION
The early years of web search (1995-1998) were characterised by innovation as researchers came to discover some of the shortcomings of traditional term-based information retrieval techniques in the face of large-scale, heterogeneous web content, and in the face of queries from users who were far from search experts. While traditional term-based matching techniques played an important role in result selection, they were not sufficiently robust when it came to delivering a reliable and relevant ranking of search results. The significant breakthrough that led to modern web search engines came about through the work of Brin and Page [1], and Kleinberg [6], highlighting the importance of link connectivity when it came to understanding the importance of web pages. In the end, ranking metrics based on this type of connectivity data came to provide a key signal for all of today's mainstream search engines.

By and large the world of web search has remained relatively stable over the past decade or more. Mainstream search engines have innovated around the edges of search but their core approaches have remained intact. However there are signs that this is now changing and it is an interesting time in the world of mainstream web search, especially as all of the mainstream players look to the world of social networks to provide new types of search content and, importantly in this paper, new sources of ranking signals. There is now considerable interest in the concept of social search, based on the idea that information in our social graphs can be used to improve mainstream search. For example, the HeyStaks system [19] has been developed to add a layer of social search onto mainstream search engines, using recommendation techniques to automatically suggest results to users based on pages that members of their social graphs have found to be interesting for similar queries in the past. HeyStaks adds collaboration to conventional web search and allows us to benefit from the past search histories of people we trust and on topics that matter to us.

In this paper we examine the role of reputation in HeyStaks' recommendation engine. Previously we have described how to estimate the reputation of a searcher by analysing how frequently their past search efforts have translated into useful recommendations for other users [9, 11]. We have also examined user behaviour in HeyStaks, and highlighted the potential for reputation to unearth users who have gained the most benefit from the system and whose activity benefits others [10]. For example, if my previous searches (and the pages that I find) lead to result recommendations to others that are regularly acted on (selected, tagged, shared etc.), then my reputation should increase, whereas if my past search efforts rarely translate into useful recommendations then my reputation should decline. In this paper we expand on previous work by considering a number of user reputation models, showing how these models can be used to estimate result reputation, and comparing the ability of these models to influence recommendation quality based on recent live-user data.

2. RELATED WORK
Recently there has been considerable interest in reputation systems to provide mechanisms to evaluate user reputation and inter-user trust across a growing number of social web and e-commerce applications. For example, the reputation system used by eBay has been examined by Jøsang et al. [5] and Resnick et al. [16]. 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 found that using information received directly from users to calculate reputation is not without its problems [16]. 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. Jøsang confirms this, stating this may lead to malicious use of the system and as such needs manual curation.

The work of O’Donovan and Smyth [14] addresses reputation in recommender systems. In this case, a standard collaborative filtering algorithm is modified to add a user-user trust score to complement 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. It is posited 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%

Other recent research has examined reputation systems employed in social networking platforms. Lazzari performed a case study of the professional social networking site Naymz [8]. 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 in [5], Lazzari 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.

The previous section outlines our intention to present different reputation models to be applied to HeyStaks users. These models are in part derived from constructing a graph based on collaborations that occur in the HeyStaks community. Perhaps two of the most well-known link analysis algorithms that are applied to online social network graphs are PageRank and HITS.

PageRank is the well known algorithm employed by the Google search engine to rank web search results [1]. The key intuition behind PageRank is that pages on the web can be modeled as vertices in a directed graph, where the edge set is determined by the hyperlinks between pages. PageRank leverages this link structure to produce an estimate of a relative importance of web pages, with inlinks from pages seen as a form of recommendation from page authors. Important pages are considered to be those with relatively large number of inlinks. Moreover, pages that are linked to by many other important pages receive higher ranks themselves. PageRank is a recursive algorithm, where the ranks of pages are a function of the ranks of those pages that link to them.

The HITS algorithm [6] was also developed to rank web search results and, like PageRank, makes use of the link structure of the web to perform ranking. In particular, HITS computes two distinct scores for each page: an authority score and a hub score. The former provides an estimate of the value of a page’s content while the latter measures the value of its links to other pages. Pages receive higher authority scores if they are linked to by pages with high hub scores, and receive higher hub scores if they link to many pages with high authority scores. HITS is an iterative algorithm where authority and hub scores are computed recursively.

A lot of work has been done in the area of link analysis in the social web space in the recent past, often by employing the techniques introduced by Page and Kleinberg. For example the well-known algorithm FolkRank [4], an adaptation of PageRank, looks to exploit users’ disposition for adding metadata to online content in order to construct a graph based on social tagging information. Work by Schiffanella et al. [18] expands on the idea behind FolkRank, and claims that examination of folksonomy data can help in predicting links between people in the social network graphs of Flickr and Last.fm.

In this paper we consider reputation models in the context of the HeyStaks social search service which seek to capture the quality of search knowledge that is contributed by users. Further, we present a framework in which user reputation is employed to influence the recommendations that are made by HeyStaks. Using data from a live-user trial, we show how this approach leads to significant improvements in the ranking of recommendations from a quality perspective. This differs from our approach in that we wish to leverage the HeyStaks social graph to determine who provides the best quality content as determined by their community.

3. THE HEYSTAKS RECOMMENDATION ENGINE

In this section we review the HeyStaks recommendation engine to provide sufficient context for this work. Further details can be found in [19] (which focuses on the relevance model) and in [11] (which focuses on the reputation model).

3.0.1 Profiling Stak Pages

Each stak in HeyStaks captures the search activities of its stak members. The basic unit of stak information is a result (URL) and each stak (S) is associated with a set of results, S = {r1, ..., rk}. Each result is also anonymously associated with a number of implicit and explicit interest indicators, based on the type of actions (for example, selecting, voting, tagging and sharing) that users can perform on these pages.

These actions can be associated with a degree of confidence that the user finds the page to be relevant. Each result page r_i in stak S, is associated with relevance indicators: the number of times a result has been selected (SI), the query terms (q1, ..., q_n) that led to its selection, the terms contained in the snippet of the selected result (s1, ..., s_k), the number of times a result has been tagged (T_j), 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 (Sh) as per Equation 1.

\[ r_i = \{q_1, ..., q_n, s_1, ..., s_k, t_1, ..., t_m, v^+, v^-, SI, T_j, Sh\} \]  

Importantly, this means each result page is associated with a set of term data (query and/or tag terms) and a set of usage data (the selection, tag, share, and voting count). The term data 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.

3.0.2 Recommending Search Results

At search time, the searcher’s query q and current stak S are used to generate a list of recommendations. Here we
discuss recommendation generation from the current stak $S$
only, although recommendations may also come from other
staks that the user has joined or created. There are two key
steps when it comes to generating recommendations. First,
a set of recommendation candidates are retrieved from $S$
based on the overlap between the query terms and the terms
used to index each recommendation (query, snippet, and
tag terms). These recommendations are then filtered and
ranked. Results that do not exceed certain activity thresholds are eliminated; such as, for example, results with only a
single selection or results with more negative votes than
positive votes (see [19]). Remaining recommendation candid-
dates are then ranked according to a weighted score of its
relevance and reputation (Equation 2), where $w$ is used to
adjust the relative influence of relevance and reputation.

\[
score(r, q) = w \times rep(r, t) + (1 - w) \times rel(q, r)
\]

(2)

The relevance of a result $r$ with respect to a query $q$ is
computed using TF-IDF [17], which gives high weights to
terms that are popular for a result $r$ but rare across other
stak results, thereby serving to prioritise results that match
distinguishing index terms, as per Equation 3.

\[
rel(q, r) = \sum_{t \in q} tf(t, r) \times idf(t)^2
\]

(3)

The reputation of a result $r$ at time $t$ ($rep(r, t)$) is an or-
thogonal measure of recommendation quality. The intuition
is that we should prefer results that originate from more rep-
utable stak members. We explore user reputation and how it
can be computed in the next section.

4. REPUTATION MODELS FOR SOCIAL
SEARCH

For HeyStaks, searchers themselves play a crucial role in
determining what gets recommended and to whom, and so
the quality of these searchers can be an important factor
to consider during recommendation. Recommendation can-
didates originating from the activities of very experienced
users, for example, might be considered ahead of candi-
dates that come from the activity of less experienced users.
This is particularly important given the potential for mali-
cious 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. This type of gaming has the potential to sig-
ificantly degrade recommendation quality; see also recent
related research on malicious users and robustness by the
recommender systems community [3, 7, 13, 15]. For this
reason we propose to complement the relevance of a page,
during recommendation, with an orthogonal measure of rep-
utation to reflect the predicted quality of the users who are
responsible for this recommendation. In fact we propose a
variety of reputation models and in Section 5 we evaluate
their effectiveness in practice.

4.1 Search, Collaboration, and Reputation

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.

![Diagram](image)

Figure 1: Collaboration and reputation: (a) the consumer $c$ selects result $r$, which has been recom-
ended based on the producer $p$’s previous activity, so that $c$ confers some unit of reputation ($rep$) on $p$.
(b) The consumer $c$ selects a result $r$ that has been produced by several producers, $p_1, ... , p_k$. Reputation
is shared amongst these producers with each user receiving an equal share of $rep$/$k$ units of reputation.

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 the quality of shared search knowledge can be estimated by looking at the
search collaborations that naturally occur 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 ac-
tion on this result caused it to be added to the search stak,
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 models is that this act of implicit
collaboration between producer and consumer confers some
unit of reputation on the producer (Figure 1(a)). And the
reputation models that we will present in what follows differ
in the way that they distribute and aggregate reputation
among these collaborations.

4.2 Graph-Based Reputation Models

We can treat the collaborations that occur among HeyStaks
users as a type of graph. Each node represents a unique
user and the edges represent collaborations between pairs of
users. These edges are directed to reflect the producer/consumer
relationships and reputation flows along these edges, and is
aggregated at the nodes. As such, the extent to which users
collaborate (i.e., the number of times each user is a producer
in a collaboration event) is used to weight the nodes in the
collaboration graph. We now present a series of graph-based
reputation model alternatives.

4.2.1 Reputation as a Weighted Count of Collabora-
tion Events

Our first and simplest reputation model calculates the rep-
utation of a producer as a weighted sum of the collaboration
events in which they have participated. The simplest case is
captured by Figure 1(a) where a single producer participates
in a collaboration event with a given consumer and benefits
from a single unit of reputation as a result. More generally however, at the time when the consumer acts (selects, tags, votes etc.) on the promoted result, there may be 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. Alternatively, a collaboration event can have a knock-on effect, where the original producer–consumer relationship is broadened as more people act on the same recommendation over time. The original consumer becomes a second producer as a new user acts on the same recommendation, and so on. Thus we need to be able to share reputation across these different producers; see Figure 1(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 \( \text{producers}(r, t) \) as per Equation 4 such that each \( p_i \) denotes a specific user \( u_i \) in a specific stak \( S_j \),

\[
\text{producers}(r, t) = \{p_1, \ldots, p_k\}. \tag{4}
\]

Then, for each producer of \( r, p_i \), we update its reputation as in Equation 5. In this way reputation is shared equally among its \( k \) contributing producers.

\[
\text{rep}(p_i, t) = \text{rep}(p_i, t-1) + 1/k. \tag{5}
\]

As it stands this reputation model is 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 \( n_c(p_i, t-1) \) 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 user \( u_i \) and a specific stak \( S_j \). Further, let \( n_c(p_i, t-1) \) be the number of these results that have been subsequently recommended and consumed by other users. We define the consumption ratio according to Equation 6; \( \kappa \) 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 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.

\[
\text{consumption}\_\text{ratio}(p_i, t) = \kappa + \frac{n_c(p_i, t-1)}{n_c(p_i, t-1)}. \tag{6}
\]

Thus, given the selection of a result \( r \) by a consumer \( c \) at time \( t \); if \( p_1, \ldots, p_k \) are the contributing producers, then we can use their consumption ratios as the basis for sharing reputation according to Equation 7.

\[
\text{rep}(p_i, t) = \text{rep}(p_i, t-1) + \sum_{p_j \in \text{producers}(r, t)} \text{consumption}\_\text{ratio}(p_i, t). \tag{7}
\]

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.

### 4.2.2 Reputation as PageRank

The PageRank algorithm can be readily applied to compute the reputation of HeyStaks users, which take the place of web pages in the graph. When a collaboration event occurs, directed links are inserted from the consumer (i.e. the user who selects or votes etc. on the recommended page) to each of the producers (i.e. the set of users whose previous activity on the page caused it to be recommended by HeyStaks). Once all the collaboration events up to some point in time, \( t \), have been captured on the graph, the PageRank algorithm is then executed and the reputation (PageRank) of each user \( p_i \) at time \( t \) is computed as:

\[
PR(p_i) = \frac{1 - d}{N} + d \sum_{p_j \in \text{producers}(p_i)} \frac{PR(p_j)}{|L(p_j)|}, \tag{8}
\]

where \( d \) is a damping factor, \( N \) is the number of users, \( M(p_i) \) is the set of inlinks (from consumers) to (producer) \( p_i \), and \( L(p_j) \) is the the set of outlinks from \( p_j \) (i.e. the other users from whom \( p_j \) has consumed results). In this paper, we use the JUNG (Java Universal Network/Graph) Framework (http://jung.sourceforge.net/) implementation of PageRank.

### 4.2.3 Reputation as HITS

As with PageRank, we use the collaboration graph and the HITS algorithm to estimate user reputation. In this regard, it seems appropriate to consider producers as authorities and consumers as hubs. However, as we will discuss in Section 5, hub scores are useful when it comes to identifying a particular class of users which act both as useful consumers and producers of high quality search knowledge. Thus we model user reputation using both authority and hub scores, which we compute using the JUNG implementation of the HITS algorithm. Briefly, the algorithm operates as follows. After initialisation, repeated iterations are used to update the authority (\( \text{auth}(p_i) \)) and hub scores (\( \text{hub}(p_i) \)) for each user \( p_i \). At each iteration, authority and hub scores are given by:

\[
\text{auth}(p_i) = \sum_{p_j \in M(p_i)} \text{hub}(p_j) \tag{9}
\]

\[
\text{hub}(p_i) = \sum_{p_j \in L(p_i)} \text{auth}(p_j) \tag{10}
\]

where as before \( M(p_i) \) is the set of inlinks (from consumers) to (producer) \( p_i \), and \( L(p_j) \) is the set of outlinks from \( p_i \) (i.e. the other users from whom \( p_j \) has consumed results).

### 4.3 Reputation and Result Recommendation

In the previous sections we have described reputation models for users. Individual stak members accumulate reputation when results that they have added to the stak 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.

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?

Before the reputation of a page is calculated, the reputation score of each producer is normalized according to the maximum user reputation score existing in the stak at the time that the recommendation is made. But how can we calculate the reputation of a page based on that of its 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, but 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 11. 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 and so their low reputations are not so much of a concern in the face of highly reputable producers.

\[
rep(r, t) = \max_{p_i \in \{p_1, \ldots, p_k\}} \left( rep(p_i, t) \right).
\]  

Equation 11

Now we have two ways to evaluate the appropriateness of a page for recommendation — the relevance of the page as per Equation 3 and its reputation as per Equation 11 — and we can combine these two scores using a simple weighted sum according to Equation 2 to calculate the rank score of a result page \( r \) and its producers \( p_1, \ldots, p_k \) at time \( t \), with respect to query \( q \).

5. EVALUATION

In previous work [19] we have demonstrated how the standard relevance-based recommendations generated by HeyStaks can be more relevant than the top ranking results of Google. In this work we wish to compare HeyStaks’ relevance-based recommendation technique to an extended version of the system that also includes reputation. In more recent prior work, our initial proof-of-concept reputation model has been outlined and motivated, and a preliminary evaluation of reputation scores assigned to early adopters of the HeyStaks system was carried out [11]. We have also showed that user reputation scores can be used to positively influence HeyStaks recommendations [12], however this work focused on only one model.

The purpose of this paper has been to build on previous work by proposing a number of alternatives to estimating the reputation of users (producers) who are helping other users (consumers) to search within the HeyStaks social search service. The aim is to explore known link-analysis techniques to find a mechanism that best captures HeyStaks users’ reputation in terms of the quality of content they provide their community. We measure each model’s effectiveness by allowing the scores to influence recommendations made by HeyStaks: The hypothesis is that by allowing reputation, as well as relevance, to influence the ranking of result recommendation, we can improve the overall quality of search results. In this section we evaluate these reputation models using data generated during a recent closed, live-user trial of HeyStaks, designed to evaluate the utility of HeyStaks’ brand of collaborative search in fact-finding information discovery tasks.

5.1 Dataset and Methodology

Our live-user trial involved 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. Each student received the same set of questions which were randomly presented to avoid any ordering bias. The questions were selected for their obscurity and difficulty; see Table 1 for a sample of these questions. Each user was allocated a desktop computer with the Firefox web browser and HeyStaks’ toolbar pre-installed; they were permitted to use Google, enhanced by HeyStaks functionality, as an aid in the quiz. 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 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. During the 60 minute trial, 3,124 queries and 1,998 result activities (selections, tagging, voting, popouts) were logged, and 724 unique results were selected. During the course of the trial, result selections — the typical form of search activity — dominated over HeyStaks-specific activities such as tagging and voting. On average, across all staks, result selections accounted for just over 81% of all activities, with tagging accounting for just under 12% and voting for 6%.

In recent work we described the performance results of this trial showing how larger groups tended to benefit from the increased collaboration effects of HeyStaks [9]. Members of shared staks answered significantly more questions correctly, and with fewer queries, than the members of solitary staks who did not benefit from collaboration. In this paper we are interested in exploring reputation. No reputation model was used during the live-user trial and so recommen-

Table 1: A sample of the user-trial questions.

<table>
<thead>
<tr>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Who was the last Briton to win the men’s singles at Wimbledon?</td>
</tr>
<tr>
<td>2. Which Old Testament book is about the sufferings of one man?</td>
</tr>
<tr>
<td>3. Which reporter fronted the film footage that sparked off Band Aid?</td>
</tr>
<tr>
<td>4. Which space probes failed to find life on Mars?</td>
</tr>
</tbody>
</table>
dations were ranked based on relevance only. However the data produced makes it possible for us to replay the user trial so that we can construct our reputation models and use them to re-rank HeyStaks recommendations. We can retrospectively test the quality of re-ranked results versus the original ranking against a ground-truth relevance; since as part of the post-trial analysis, each selected result was manually classified as relevant (the result contained the answer to a question), partially relevant (the result referred to an answer, but not explicitly), or not-relevant (the result did not contain any reference to an answer) by experts.

5.2 User Reputation

We now examine the type of user reputation values that are generated from the trial data. In Figure 2, box-plots are shown for the median reputation scores across the 4 shared staks and for each reputation model. Here we see that for the WeightedSum model there is a clear difference in the median reputation score for members of the 5 person stak when compared to members of the larger staks. This is not evident in results for the PageRank model, which shows very similar reputation scores, regardless of stak size. For the Hubs and Authority models we see very exaggerated median reputation scores for the largest 25-person stak, whereas the median reputation scores for members of the smaller staks are orders of magnitude less. Next we consider, for members of each stak, how the reputation scores produced by the four reputation models compare. The pairwise rank correlations between user reputation scores given by each reputation model are shown in Table 2. With the exception of the 5 person stak (likely due to the relatively small number of users in this particular stak), correlations are seen to be high between the WeightedSum, PageRank and Authority models. For example, pairwise correlations between these models in the range 0.90-0.94 are observed for the 25 person stak. In contrast, the correlations between the Hubs model and the other models are much lower; and indeed, are negative for the smaller 5 and 9 person staks. It is difficult to draw precise conclusions about the Hubs correlations for each of the staks concerned (given the constrained nature of the user-trial and the different numbers of users in each stak), but since the HITS Hubs metric is designed to identify pages that contain useful links towards authoritative pages in the web search domain (analogous to good consumers rather than producers in our context), such low correlations are to be expected with the other models which more directly focus on producer activity.

Further, a desirable property of a reputation model is that it should capture consumption diversity, meaning that in order for producers to gain high reputation, many consumers should benefit from the content that producers contribute to staks. Table 3 shows the Pearson correlation between the number of distinct consumers per producer (per stak) and producer reputation according to each of the four reputation models tested. Across all staks, Authority displays the highest correlations (between 0.98 and 1), indicating that this model is particularly effective in capturing consumption diversity. This is to be expected, given that user Authority scores are directly influenced by the number of consumers interacting with them. In contrast and given the nature of the Hubs model, it unsurprisingly fails to capture consumption diversity. For the larger staks, we can see good correlations are achieved for the WeightedSum and PageRank models also, but less so for the smaller staks. In future work, we plan on refining our WeightedSum model in order to better reflect consumption diversity for such small-sized staks.

Figure 2 shows that there are significant differences in user reputation scores produced by the four different models. But how best to interpret these differences? In this work, we consider that the true test of these reputation models is the extent to which they improve in the quality of results recommended by HeyStaks. We have described how HeyStaks combines term-based relevance and user reputation to generate its recommendation rankings (see Equation 2); in the following section we regenerate each of the recommendation lists produced during the trial using our reputation models and compare the performance of each.

5.3 From Reputation to Quality

Since we have ground-truth relevance information for all of the recommendations (relative to the quiz questions), we can then determine the quality of the resulting recommendations. Specifically, we focus on the top recommended result and note whether it is relevant (that is, contains the answer to the question) or not relevant (does not contain the answer to the question). For each reputation model we compute an overall relevance rate, as the ratio of the percentage of recommendations (relative to the quiz questions), we can then determine the quality of the resulting recommendations made by the standard HeyStaks ranking (i.e. when \( w = 0 \) in Equation 2) in the

<table>
<thead>
<tr>
<th>Stak Size</th>
<th>5</th>
<th>9</th>
<th>19</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeightedSum</td>
<td>0.41</td>
<td>0.52</td>
<td>0.78</td>
<td>0.85</td>
</tr>
<tr>
<td>PageRank</td>
<td>0.75</td>
<td>0.68</td>
<td>0.85</td>
<td>0.92</td>
</tr>
<tr>
<td>Hubs</td>
<td>-0.86</td>
<td>-0.63</td>
<td>0.43</td>
<td>0.26</td>
</tr>
<tr>
<td>Authority</td>
<td>1.00</td>
<td>1.00</td>
<td>0.98</td>
<td>0.99</td>
</tr>
</tbody>
</table>

Table 3: Correlations between the number of distinct consumers per producer per stak and producer reputation.
However, given the man-
lect, tag, vote etc. HeyStaks recommendations deriving from
extent to which users are good
frequently selected by other users), while
ity search knowledge (i.e. users whose recommendations are
capture the extent to which users are good
model would outperform
text. One might expect, for example, that the
an interesting property of the HITS algorithm in this con-
achieve similar mean relevance benefits of between 21-25%.
the other models across all values of
such that a rel-
relevance benefit
weighting (w) used in Equation 2 to adjust the influence of
term-based relevance versus user reputation during recom-
mendation. The results for all four reputation models indic-
ate a significant benefit in recommendation quality when
compared to the standard HeyStaks recommendations. As
we increase the influence of reputation over relevance dur-
ing recommendation (by increasing w) we see a consistent in-
crease in the relevance benefit, up to values of w in the range
0.5-0.7. For example, we can see that for w = 0.5, the reputa-
tion models are driving a relative improvement in recom-
mendation relevance of about 30-40% compared to default
HeyStaks’ relevance-only based recommendations. Overall
the Hubs model performs best. It consistently outperforms
the other models across all values of w and achieves a max-
imum relevance benefit of about 45% at w = 0.7. Looking
at mean relevance benefit across reputation models, Hubs
is clearly the best performer. For example, Hubs achieves
a mean relevance benefit of 31%, while the other models
achieve similar mean relevance benefits of between 21-25%.
In a sense, this finding is counter-intuitive and highlights
an interesting property of the HITS algorithm in this con-
text. One might expect, for example, that the Authority
model would outperform Hubs, given that Authority scores
capture the extent to which users are good producers of
quality search knowledge (i.e. users whose recommendations are
frequently selected by other users), while Hubs captures the
extent to which users are good consumers (i.e. users who
select, tag, vote etc. HeyStaks recommendations deriving from
the activity of good producers). However, given the man-
n in which the collaboration graph is constructed (Section
4.2), once a user has consumed a recommended result, then
that user is also considered to be a producer of the result in
question if it is recommended by HeyStaks and selected by
other users at future points in time. Thus, good consumers
— who select recommended results from many good produ-
ders (i.e. producers with high Authority scores) — serve
as a “filter” for a broad base of quality search knowledge, and
hence re-ranking default HeyStaks recommendations using
reputation scores from the Hubs model leads to the better
recommendation performance observed in Figure 3.

5.4 Limitations
In this evaluation we have compared a number of reputa-
tion models based on live-user search data. One limitation
of this approach is that although the evaluation uses live-
user search data, the final recommendations are not them-
selves evaluated using live-users. Instead we replay users’
searches to generate reputation-enhanced recommendations.
The reason for this is the difficulty in securing sufficiently
many live-users for a trial of this nature, which combines
a number of reputation models and therefore a number of
experimental conditions. That being said, our evaluation
methodology is sound since we evaluate the final recommen-
dations with respect to their ground-truth relevance. We
have an objective measure of page relevance based on the
Q&A nature of the trial and we use this to evaluate the gen-
une relevance of the final recommendations. The fact that
our reputation models deliver relevance benefits above and
beyond the standard HeyStaks recommendation algorithm is
a clear indication that reputation provides a valuable rank-
ing signal. Of course this evaluation cannot tell whether
users will actually select these reputation ranked recommend-
dations, although there is no reason to suppose that they
would treat these recommendation differently from the de-
fault HeyStaks recommendations, which they are inclined to
select. We view this as a matter for future work.

Another point worth noting is that the live-user trial is
limited to a specific type of search task, in this case a Q&A
search task. Although such a task is informational in nature
(according to stipulations set out by Broder [2]) it would
be unsafe to draw general conclusions in relation to other
more open-ended search tasks. However, this type of focused
search task is not uncommon among web searchers and as
such we feel it represents an important and suitable use-
case that is worthy of evaluation. Moreover, previous work
[19] has looked at the role of HeyStaks in more open-ended
search tasks to note related benefits to end-users from its
default relevance-based recommendations. As part of our
future work we are currently in the process of deploying and evaluating our reputation model across similar general-purpose search tasks.

6. CONCLUSIONS

In this paper we have described a number of different user reputation models designed to mediate result recommendation in collaborative search systems. We have described the results of a comparative evaluation in the context of real-user data which highlights the ability of these models to improve overall recommendation quality, when combined with conventional recommendation ranking metrics. Moreover, we have found that one model, based on the well-known HITS Hubs metric seems to perform especially well, delivering relative improvements of up to 45%. We believe that this work lays the ground-work for future research in this area which will focus on scaling-up the role of reputation in HeyStaks and refining the combination of relevance and reputation during recommendation.

Our reputation model is utility-based [11], based on an analysis of the usefulness of producer recommendations during collaboration events. Currently, in HeyStaks the identity of users (producers and consumers) is not revealed and so users do not see where their recommendations come from. In the future it may be appropriate to relax this anonymity condition in certain circumstances (under user control). By doing so it will then be possible to individual users to better understand the source of their recommendations and the reputation of their collaborating users. As such this model can ultimately lead to the formation of trust-based relationships via search collaboration.

7. ACKNOWLEDGMENTS

This work is supported by Science Foundation Ireland under grant 07/CE/I1147.

8. REFERENCES


