<table>
<thead>
<tr>
<th>Title</th>
<th>A model of collaboration-based reputation for social recommender systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authors(s)</td>
<td>McNally, Kevin</td>
</tr>
<tr>
<td>Publication date</td>
<td>2014</td>
</tr>
<tr>
<td>Publisher</td>
<td>University College Dublin. School of Computer Science and Informatics</td>
</tr>
<tr>
<td>Link to online version</td>
<td><a href="http://dissertations.umi.com/ucd:10029">http://dissertations.umi.com/ucd:10029</a></td>
</tr>
<tr>
<td>Item record/more information</td>
<td><a href="http://hdl.handle.net/10197/6790">http://hdl.handle.net/10197/6790</a></td>
</tr>
</tbody>
</table>
A MODEL OF COLLABORATION-BASED REPUTATION FOR SOCIAL RECOMMENDER SYSTEMS

KEVIN McNALLY

A thesis submitted to University College Dublin in fulfillment of the requirements for the degree of Philosophiae Doctor (PhD)

School of Computer Science and Informatics

Supervisor: Prof. Barry Smyth
Secondary Supervisor: Dr. Michael P. O’Mahony
Head of School: John Dunnion

September 2013
ABSTRACT

Today’s online world is one full of rich interactions between its users. In the early days of the web, activity was almost exclusively solitary, now however, users regularly collaborate with one another, often mediated by a piece of content or service. In offline communities, continued good behaviour and long-term relationship building leads naturally to good reputation, however online users often remain anonymous to their community and so trust building can be difficult to foster among community members. As such there has arisen a need for the system to calculate user reputation itself. Now, online reputation systems provide a variety of benefits to the platforms that employ them such as an incentive mechanism for good behaviour and improving the robustness of the platform. However, these systems are often based on ad-hoc activity metrics, and thus do not generalise to multiple platforms or different tasks.

In this thesis we introduce a novel approach to capturing and harnessing online reputation. Our approach is to develop a computational model of reputation that is based on the various types of collaboration events that naturally occur in many different types of online social platforms. We describe how a graph-based representation of these collaboration events can be used to aggregate reputation at the user-level and we evaluate a variety of different aggregation strategies. Further, we show how the availability of this type of user reputation can be used to influence traditional recommender systems by combining relevance and reputation at recommendation time. A major part of our evaluation involves integration with the HeyStaks social search system, testing our approach on real-user data from the service.
The thesis is based on the following articles and conference papers, published during the course of my PhD:

1. **Towards a Reputation-based Model of Social Web Search**
   Kevin McNally, Michael P. O’Mahony, Barry Smyth, Maurice Coyle and Peter Briggs
   Published in *IUI ’10: Proceedings of the 15th International Conference on Intelligent User Interfaces* (2010), pp 179–188

2. **A Case-study of Collaboration and Reputation in Social Web Search**
   Kevin McNally, Michael P. O’Mahony, Barry Smyth, Maurice Coyle and Peter Briggs
   Published in *ACM TIST: Transactions on Intelligent Systems Technology* 3.1 (2011): 4

3. **Evaluating User Reputation in Social Web Search**
   Kevin McNally, Michael P. O’Mahony, Barry Smyth
   Published in *RSWeb ’11: 3rd Workshop on Recommender Systems and the Social Web, in association with the 5th ACM Conference on Recommender Systems* (2011)

4. **Models of Page Reputation in Social Search**
   Kevin McNally, Michael P. O’Mahony and Barry Smyth

   Kevin McNally, Michael P. O’Mahony and Barry Smyth
   Published in *User Modeling and User-adapted Interaction* (2013), pp 1–42
Kevin McNally, Michael P. O’Mahony and Barry Smyth
Published in ICWSM ’13: Proceedings of the 7th AAAI Conference on Weblogs and Social Media (2013)
ACKNOWLEDGEMENTS

I have been extremely lucky as a postgraduate student to have had support from so many people, all of whom are deserving of much more than simply a message of thanks in this thesis.

Firstly, I wish to thank my supervisor Prof. Barry Smyth for providing guidance and support from my very first day in his research group, and for instilling an enthusiasm and confidence that four years ago I did not know I could have. I am indebted to HeyStaks’ Maurice Coyle and Peter Briggs for their always-open door, and for fielding any and all questions a new PhD student might have. I would also like to thank my Clarity Centre colleagues Owen, John, Sandra, Steven, Ruihai, Zurina, Rachael and Kevin. It was a pleasure to work with each and every one of you.

Special thanks is reserved for my mentor Dr. Michael O’Mahony. Mike displayed unending patience with me throughout my PhD and was a constant source of useful knowledge that was crucial to me finishing the work presented here.

As always, I am immensely appreciative for the support of my family and friends, in particular my parents Claire and Desie, and my sisters Gemma and Maeve. Your constant encouragement, support, and willingness to pick me up and dust me off when I most needed it has made all this possible.

The work presented in this thesis is supported by Science Foundation Ireland under grant 07/CE/I1147.
# CONTENTS

1 INTRODUCTION ................................................. 1
   1.1 The Evolving Web ...................................... 2
   1.2 The Age of Search ..................................... 2
   1.3 Social Web and the Age of Collaboration .......... 10
   1.4 Towards A Reputation Web ......................... 16
       1.4.1 Defining Online Reputation .................. 17
       1.4.2 Feedback-based Reputation ................. 19
       1.4.3 Behaviour-based Reputation ............... 22
       1.4.4 Online Reputation: A Summary ............ 24
   1.5 Application Case-study: The HeyStaks Social Search Utility . 25
   1.6 Core Contributions .................................... 31
       1.6.1 A Computational Model of Reputation ...... 31
       1.6.2 From Reputation to Recommendation ..... 32
       1.6.3 Generalising the Approach ................. 32

i A COMPUTATIONAL MODEL OF REPUTATION .......... 33

2 PAPER 1: TOWARDS A REPUTATION-BASED MODEL OF SOCIAL WEB SEARCH 35

3 PAPER 2: A CASE-STUDY OF COLLABORATION AND REPUTATION IN SOCIAL WEB SEARCH 47

ii FROM USER REPUTATION TO ITEM REPUTATION FOR RECOMMENDATION .......... 81

4 PAPER 3: EVALUATING USER REPUTATION IN SOCIAL WEB SEARCH 83
5 PAPER 4: MODELS OF PAGE REPUTATION IN SOCIAL SEARCH 93
6 PAPER 5: A COMPARATIVE STUDY OF REPUTATION MODELS FOR SOCIAL RECOMMENDER SYSTEMS 103

iii GENERALISING THE APPROACH ......................... 145

7 PAPER 6: A MODEL OF COLLABORATION-BASED REPUTATION FOR THE SOCIAL WEB 147
8 CONCLUSION .................................................. 153
8.1 Summary of Contributions ........................................ 153
  8.1.1 A Computational Model of Reputation .................... 153
  8.1.2 From User Reputation to Item Reputation for Recommendation 154
  8.1.3 Generalising the Approach .................................. 155
8.2 Summary of Limitations .......................................... 155
8.3 Future Directions and Open Questions ....................... 156
  8.3.1 Beyond Social Search .................................... 156
  8.3.2 Integration and Scalability ............................... 158
  8.3.3 Reputation and Robustness ............................... 159

iv APPENDIX

A APPENDIX ......................................................... 163
  A.1 Paper 1 Experiment Methodology: Identification of Experimental Users 163
  A.2 Papers 2–5 Experiment Methodology: Ground-truth Validation ........ 164
  A.3 Paper 2: Misuse of the term “Significant” to Describe Experiment Findings .............................. 165

BIBLIOGRAPHY ....................................................... 167
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A high-level architecture diagram of a search engine.</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Yahoo in 1996: An early example of a Web Directory</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Altavista: A popular early search engine</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>An example graph of pages [56], with directed edges representing hyperlinks between pages. The score denoted on each page is its PageRank. The scores that flow through a page’s forward links are representative not only of its PageRank, but the number of its forward links. A Page’s PageRank is a sum of the scores that flow through its backlinks.</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>Google’s approach to web search – measuring the authority of web pages by building a graph of links – changed the face of web search in the late 1990’s.</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>An example graph of pages, with directed edges representing hyperlinks between pages. In each case here, the nodes are sized and scored according to the HITS algorithm (Hubs in (a), Authorities in (b))</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>A Facebook Graph Search result list. In this screenshot, the search is “People who like things I like”. This is an example of how users can search within a social context - exploring results based on their friends’ activities and interests.</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>An example graph snippet illustrating the structure of the Facebook graph. Nodes in the graph can represent people or pages. Directed edges represent interactions between entities. These edges are labeled according to the nature of the interaction. Some edges can be bi-directional (for example, note the edge labeled “FRIEND”).</td>
<td>12</td>
</tr>
<tr>
<td>9</td>
<td>An example Google Plus profile page, showing how a user can make use of the “circles” functionality to manually curate their information streams on the platform.</td>
<td>14</td>
</tr>
<tr>
<td>10</td>
<td>A diagram (introduced in [48]) illustrating the relationships between trust, reputation and reciprocity</td>
<td>18</td>
</tr>
</tbody>
</table>
Figure 11 eBay’s feedback-based reputation system. This screenshot shows the reputation of seller “unbeatable-uk”. 67,400 buyers have given them positive feedback, which accounts for 99.6% of the total feedback they have received. They average 5 stars across all performance indicators. eBay has rewarded this high-reputation user with a “top-seller” badge. This badge allows for buyers to easily find high-reputation sellers.

Figure 12 Aggregate review scores for a typical Airbnb host (including an example review). The star ratings are an average taken over all renters’ explicit feedback given about this host.

Figure 13 An example Peerindex profile. The influence score shown is output from a proprietary algorithm. The user’s influence is broken down by topic, according to the numbers below the profile.

Figure 14 A profile of a high-reputation Stackoverflow user, showing their reputation score. This score is public knowledge on the site.

Figure 15 An example search session in Google. The user is searching for information on Hard Rock bikes, but is not providing Google with enough context to give the kind of results that satisfy the user. Note the top result is regarding the Hard Rock Cafe.

Figure 16 Again, the user searches for information on Hard Rock bikes, this time using HeyStaks. The user has joined the “Mountain Biking” stake and, as a result, the top search results relate to bikes. These top results are HeyStaks recommendations.

Figure 17 The HeyStaks browser toolbar.

Figure 18 HeyStaks’ back-end server architecture. The reputation engine is fully integrated, generating scores for recommendation candidates based on that of the users who produced them.
| Figure 19 | Reputation of a page in HeyStaks can be calculated by translating the reputation of its producers. Note that due to the type of activity seen on the platform, all users who have acted on a page can be thought of as the producers of that page. In this case, ........................................... 30 |
| Figure 20 | Network sizes for the 26 trial users in paper 1 (see Chapter 2). 163 |
INTRODUCTION

The World Wide Web has brought about a fundamental change in how information and services are provided to society. The web has evolved from a collection of static pages to a platform that facilitates user interaction and community engagement in many deep and meaningful ways. A key feature of the early web was the hyperlink, a concept introduced by Berners-Lee et al. [5], and inspired by earlier work in different domains such as microfilm reels [9] and word processing software [49]. These logical links between pages enabled web users to traverse through different pieces of content. As more content was put online, these interconnected pages formed a large graph-like structure, which we now refer to as the web. Soon, there was so much content available online that it became difficult to not only traverse, but to find relevant, authoritative content. Innovators soon figured out ways in which to leverage the link structure of the web to find the best information for its users – from search engines [7, 29], to online social networks [35, 50] and personalised recommender systems [20, 66].

The notion that the web is simply a collection of linked static content is now a distant memory. Today, the social web is one full of rich interactions taking place between its inhabitants. The rise of social networks like Facebook\(^1\), Google+\(^2\) and Twitter\(^3\) mean that the modern web is as much about people and relationships as it is about pages and links. And, just as innovation arose from exploring the graph structure of the of pages in the early days of the web, now we examine the interaction-based web of people to answer key questions. For example, how are people behaving, collaborating, consuming and producing? This thesis seeks to leverage the link structure that occurs naturally in online social platforms in order to answer two such questions: How reputable is each user with respect to their community? And, crucially, can we use reputation information to measure the quality of content users produce?

This introductory chapter tracks the history of research and application in the area of

\(^1\) http://www.facebook.com
\(^2\) http://www.plus.google.com
\(^3\) http://www.twitter.com
online reputation, but first details innovation the early days of the web, highlighting insights from that time that are pertinent to answering these questions.

1.1 THE EVOLVING WEB

The web’s genesis occurred in 1989. Tim Berners-Lee devised the Hypertext Markup Language (HTML) and the Hypertext Transfer Protocol, which together provided people with a means to easily share content across the internet [5]. A year later, Berners-Lee created the first web browser to interpret HTML documents, and thus the World Wide Web was born. HTML was not unlike other markup languages of the day, such as SGML, but a key feature which formed the cornerstone of the web was the Hyperlink. Pages written in HTML could contain a piece of information that allowed a reader to travel to a different web page by simply clicking on highlighted text – an idea that dated back to the 1940s [9]. This aided users in discovering new pieces of content, and provided content creators with a way to make their web pages accessible.

As the volume of information produced for consumption on the web grew, so too did the need for technology to enable users to find relevant, authoritative information effectively. Efforts were made to build an index of all web search pages using automated bots to traverse the web. These indexes could then be searched over using traditional information retrieval techniques, allowing users to search over any word in any web page. Armed with the idea of marrying web page indexes with traditional information retrieval approaches, web search companies battled for popularity with an ever-growing World Wide Web user base.

The number of pages on the web soon grew in number from the thousands to the millions. Focus was put on creating a technology infrastructure that could effectively index all existing pages and deliver results to the user instantaneously. An architecture for search technology was developed, and indeed a standard model for search from this era has endured to today.

1.2 THE AGE OF SEARCH

Figure 1 illustrates a typical search engine architecture. In the mid 1990s, innovation centred on creating technology that could reasonably crawl the vast number of doc-

---

Figure 1: A high-level architecture diagram of a search engine.

Architectures that took this approach to web search were able to handle a large volume of user queries: at its peak, Altavista was serving around 13 million queries per day [3]. However search companies soon found that simply indexing pages according to its content in order to rank against an input query was not an effective way to handle an ever growing web. It became impossible for users to reasonably digest all information on a given topic that was returned by the search engines of the
As such there came a need for technology to aid them in finding the information they needed. An early approach to addressing this issue was the web directory. Such sites aimed to organise links to web pages in a uniform way that enabled users to traverse the link structure to find what they needed. An early example was Yahoo!, which consisted of a single page with links to broad, searchable categories. Figure 2 shows an early Yahoo! search interface. However, this user experience belied the popularity of other web search sites that offered a simpler, cleaner user interface. For example, Altavista were early adopters of the single search bar on an otherwise blank screen; see Figure 3. However, due to the sheer size of the web, even in the 1990s, a single search query in Altavista would return hundreds of search pages.

Web search companies at the time found that traditional information retrieval approaches could not rank search pages in a way that satisfied the user. Often the web page that was the best match to a user’s query was not the most useful or authoritative content available. According to Brin and Page [7], in 1997 web search efficiency had reached such a low that only one of the top four engines at the time would display
their own web site on a results page when their name was entered as a search query. A need arose for search engines to find both the most authoritative web pages as well as those most relevant to a user’s query. That way, pages could be ranked effectively, displaying content that was both relevant and useful to the user. The late 1990’s brought a seminal innovation in web search. A number of researchers were inspired by work analysing the authority of research papers in citation graphs and found that analysing hyperlinks was a useful way of identifying the importance of web pages. Related concepts such as Eigenvector centrality, which measures influence of nodes in a network based on the relative score of its in-linking nodes, inspired a new way of thinking in web search. In 1998, Page et al. [56] proposed a solution based on the idea that a web page’s importance is related to the importance of other web pages which link to it. A similar idea was proposed by Kleinberg [29], who refers to leveraging the link structure of the web to find both authoritative pages, and “hub pages” that provide infrastructure to the web. It appeared that links were the key to measuring the authority of web content.
The work of Page et al. [56] shows that pages on the web can be modeled as vertices in a directed graph, in which hyperlinks are represented as edges. Their algorithm, known as PageRank, examines this structure in order to calculate the relative importance of each page, assigning each page a score that reflects their position of importance in the graph. The algorithm itself is recursive, so a page’s score (or simply “PageRank”) is dependent not just on the number of in-linking pages, but the PageRank of those pages. This is based on the notion that a hyperlink to another page is an endorsement of that page, and that important web pages will link to other important web pages. Imagine a scenario where there exists a set of web pages $N$, a particular page $p_i$, and its corresponding set of in-linking pages $M(p_i)$ (not including itself, if that in-link exists). Its PageRank (PR) is computed as follows:

$$PR(p_i) = \frac{1 - d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{|L(p_j)|},$$

where $PR(p_j)$ is the PageRank of an in-linking page, $L(p)$ is the number of out-links on that page, and $d$ is a damping factor to ensure no page gains too much influence, and to assign all pages at least a small default score, even if a page receives no in-links. This means that “sinks” – i.e. pages which have no forward links – have a default score that is distributed across the entire graph. Equation 1 shows that a page’s score is a simple summation over each in-linking page’s ratio of PageRank to the number of out-linking pages. The complete set of PageRanks for a given graph.
of web pages forms a probability distribution, meaning the scores of all pages sum to one. The model takes into account the “random surfer”, that is a web user who gets bored of following links in a graph of pages and decides to visit a page at random. The PageRank of a page can be viewed as a probability that a user, travelling to a page at random, will land on that particular page.

Figure 4 illustrates an example set of pages with each’s computed PageRank score. This simple graph illustrates how each page’s PageRank is dependent on that of its in-linking pages. What this can mean is that simply receiving lots of in-links does not always result in a high PageRank score, rather it is the PageRank of those in-linking pages. Google\(^5\) was the first website of its kind, using link-analysis techniques to rank search results; see Figure 5.

![Google! Beta](http://www.google.com)

Figure 5: Google’s approach to web search – measuring the authority of web pages by building a graph of links – changed the face of web search in the late 1990’s.

Kleinberg [29] proposed a similar algorithm – known as Hyperlink-induced Topic Search (HITS) – that finds authoritative pages in the web of links. Although the work was published later, this algorithm was a precursor to PageRank. However, there are key differences between the two approaches. Unlike PageRank, HITS assigns two scores to each web page in a network: An “Authority” score that measures the page’s value to the network, and a “Hub” score that indicates the value of its links to other web pages. A page’s Authority score is determined by the Hub score of all in-linking pages, and in turn its Hub score determined by the Authority score of the pages it links to. Similar to PageRank, HITS is an iterative algorithm that assigns a default value to each page in a graph and recursively calculates each page’s Authority and

\(^5\) http://www.google.com
Hub score. In the case of HITS, every page in a network is assigned initial Hub and Authority scores of 1. Then at each iteration, the scores for page $p_i$ are updated according to the following:

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

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

Figure 6: An example graph of pages, with directed edges representing hyperlinks between pages. In each case here, the nodes are sized and scored according to the HITS algorithm (Hubs in (a), Authorities in (b)).

Where $M(p_i)$ is the set of $p_i$’s in-linking pages, and $L(p_i)$ is the set of pages $p_i$ links to. Figure 6 shows HITS computed over the same page graph as displayed in Figure 4. The similarity between the output from PageRank and HITS Authorities (shown in Figure 6 (b)) is evident, with the ranking of nodes according to their scores roughly the same. Figure 6 (a), showing pages’ HITS Hubs scores, reveals how pages can be used as a source of authoritative content, given that pages with high Hubs scores link to pages with high Authority scores.

Link-analysis approaches to calculate page importance have laid the foundations for today’s web search platforms. Since their introduction, focus has largely been
on improvement by complementing such approaches. For example, much research has examined how machine learning techniques such as Support Vector Machines and Neural Networks can be used to better rank search results for the end user [37]. Other techniques have been utilised to cluster web pages for topic extraction [31] or improved document retrieval performance [72]. Indeed, researchers have attempted relative improvements to PageRank itself. Haveliwala [22] deduce the topic being searched on by looking at query terms, and score retrieved web pages as a weight against each’s PageRank. Information retrieval methods such as query expansion [70] and query substitution [28] – where relationships between words in a set of web pages are exploited have been used to improve search engine performance.

Recent developments in other web technologies have influenced how people think about content delivery in search. The Semantic Web – an initiative that aims to connect web pages together according to the meaning of their content – can offer a new signal for search. Now pages are not just connected via hyperlinks, they are also connected via similarities as set out by some machine-learned ontology. Research by Guha et al. [19] states that semantic web information can be used to improve retrieval accuracy for users who are performing “research searches”. Indeed, the amount of semantic web information available has become so vast that there exist its own tangible information retrieval issues. Ruotsalo [60] has developed a platform for retrieval of semantic web information based on traditional information retrieval approaches.

In recent years, the experience users have on many mainstream search engines has become more personalised. If users have an account with Google, the search results they receive is tailored to them in a number of ways. For example, their history dictates the kind of results they receive in the future.7. Research speaks to the benefits of this approach to search: Pitkow et al. [57] maintain that its benefits “can be significant, appreciably decreasing the time it takes people – novices and experts alike – to find information”. Similarly, Dou et al. [16] found that using click-based information to influence results returned to the user resulted in a positive improvement in the user’s experience. However, the same research highlighted the challenges in tailoring search results. Users’ profile information, when used as a context in ranking search results, was an inconsistent performer. Casting the net wider, Teevan et al. [67] merge search history, profile information and content from other sources such as email to build a richer information source on users with which search technologies

6 http://www.w3schools.com/web/web_semantic.asp
7 http://googleblog.blogspot.ie/2009/12/personalized-search-for-everyone.html
Figure 7: A Facebook Graph Search result list. In this screenshot, the search is "People who like things I like". This is an example of how users can search within a social context - exploring results based on their friends’ activities and interests.

Users’ social interactions are now a useful context for influencing users’ search experience. “Social Search” is now a prevalent concept as today’s online search platforms begin to focus on leveraging their data to promote information discovery. This concept is discussed in detail in Section 1.5.

1.3 SOCIAL WEB AND THE AGE OF COLLABORATION

Search engines became so effective at finding relevant content that they became web users’ chief tool for discovering information online. However in the early 2000’s, search and queries gave way to social networks and sharing. The web began to be-
come a more social place, and gradually people began to use their online social networks to discover content. Public social networking services such as Friendster\(^8\), Myspace\(^9\) and Facebook gave users a public platform not only to share links to content of interest, but also to create and share content of their own. These websites dovetailed with content sharing platforms such as Blogger\(^10\) and YouTube\(^11\), allowing for an explosion in the amount of user-generated content on the web.

Today, the centre of gravity of the web has changed from pages and links to people and relationships. Billions are now connected to others in a variety of different ways, forming complex and mature social graphs. For example, as of October 2012, Facebook has one billion monthly active users, and is continuing to grow\(^12\). Similarly the microblogging service Twitter has over 200 million active users\(^13\). These users are all sharing content with their friends and followers at an unprecedented rate, for example, Facebook has seen users “like” content 2.7 billion times, share over 300 million photos and upload 500 terabytes of data for their friends’ consumption\(^14\). Analyzing how users interact in online social spaces has never been more important, as people are now collaborating in a much deeper, more meaningful way than simply clicking a link their friend posted online. Twitter has become a legitimate news medium where users can re-tweet information, disseminating it much more quickly than with traditional media. Wikipedia, itself a collaborative project relying on voluntary interaction, has become the sixth most popular site on the web today\(^15\).

People are merging the online and offline worlds to collaborate on content and services in a variety of different ways, for example, providing short-term rental solutions (Airbnb\(^16\)), coordinating human-intelligence tasks (Amazon Mechanical Turk\(^17\)), outsourcing small jobs to their real-life community (Taskrabbit\(^18\)), and crowd funding business ventures (Kickstarter\(^19\)). These kinds of activities pervade the social web

---

8 http://www.friendster.com  
9 http://www.myspace.com  
10 http://www.blogger.com  
11 http://www.youtube.com  
12 http://newsroom.fb.com/Key-Facts  
13 https://business.twitter.com/basics/what-is-twitter/  
16 http://www.airbnb.com  
17 http://www.mturk.com  
18 http://www.taskrabbit.com  
19 http://www.kickstarter.com
and show that, at its heart, the social web is about collaboration. Such an idea has been explored as a concept in its own right [6].

![Figure 8: An example graph snippet illustrating the structure of the Facebook graph. Nodes in the graph can represent people or pages. Directed edges represent interactions between entities. These edges are labeled according to the nature of the interaction. Some edges can be bi-directional (for example, note the edge labeled "FRIEND").](image)

A key area of interest in the online social space involves looking at how the information people provide about themselves on these social platforms – who their friends are, what interests they have, the kind of content they create – can be leveraged in order to improve the platform or enhance user experience. For example, Twitter use both the content you post on the site as well as your follower/following information in order to recommend other users you may wish to follow20. More recently, efforts have centered around building technology to effectively search over social graphs. Facebook has created a social search engine called “Graph Search”21 based on information generated by its user-base. The aim is to allow people to search in a social context, not just for information such as “restaurants my friends like”, but also for people that share similar interests or similar tastes (see Figure 7 for an example search query). With this new technology, Facebook is not only making use of information

21 http://www.facebook.com/about/graphsearch
it has on individual entities (friends, organizations, places etc), but the relationships between these entities. For example, Figure 8 shows a small area of a larger social graph. From examining only these two nodes, it is obvious that relationships can be multifaceted, drawn from many different types of interactions. This forms an incredibly rich information source for Facebook, who have created a platform that provides the user with a more personalized, context-sensitive and social search experience.

On other platforms, users can curate their own information or social ties. For example, on the photo-sharing site Pinterest, users can manage their own collections of photos based on their own interests. Interestingly, although they curate their own collections, the photos may come from others in the Pinterest community, and so they are implicitly linking with content providers based on individual interests. On the social platform Google+, users can categorize their friends into “circles” based on any criteria they wish. This functionality enables them to consume content from, and broadcast messages to, these circles on an individual level. Figure 9 shows an example user who has categorized their friends into a number of different circles. For example, if they only wish to see content posted by family members, the user simply needs to click on the relevant circle. Interestingly, Google allows this manual curation at the social level to pervade the user experience of its other platforms: The company sees Google+ as a “social blanket” that can add a social context to other services they provide. For example a person’s actions on their Google+ profile, such as the nature of consumed or produced content, their explicit interests and those of their friends, may well have an impact on results returned to them when querying Google Search.

Contemporary research interests have reflected this proliferation of social web technology. Research by Kwak et al. [33] suggests that online social networks that have formed on websites such as Twitter are less similar to offline networks than one might imagine. Further, they found that reach and influence are distinct, in other words if a user has many followers that does not necessarily mean their content will be re-tweeted (a strong signal of content utility). These findings prove that in order to improve upon online social web technology, a new level of understanding is required. As such, there has been a clamour to understand not just the structure of networks

23 http://www.pinterest.com
24 http://thenextweb.com/socialmedia/2012/03/08/for-the-last-time-lets-all-say-it-together-google-is-not-a-social-network/
formed by interacting users \textsuperscript{42} but how users interact as part of these networks. For example, Wilson et al. \textsuperscript{69} propose that examining relationships between users on social web platforms can lead to various insights such as user-user trust. Specifically, it is important to look beyond simple social links and look at the deeper, more meaningful interaction that occurs on these platforms.

Due to the remote nature of interaction on the social web, automatically detecting communities of people that might occur naturally on these platforms has been of great interest. Well-established ideas from other fields of research have been transferred to this new domain. Newman et al., in \textsuperscript{50} and \textsuperscript{51}, presented work that introduces a number of community detection algorithms, taking ideas from sociology (hierarchical clustering), computer science (graph partitioning) and even electronics (resistor networks) to propose solutions to effectively find communities within real-life social and collaborative networks. This work has since been extended to identify communities in much larger networks, which better reflect real-life scenar-
ios in today’s social web platforms. For example, Leskovec et al. [35] tested their approaches on data gained from a number of social websites such as Linkedin25, Epinions26 and Flickr27 and found that communities in these large networks tended to be well-defined and tightly knit.

Insights gained from examining online social networks can be leveraged to improve user experience and encourage deeper engagement with social web platforms. For example, Cai et al. [10] propose using the traditional recommender system approach known as “Collaborative Filtering” to recommend to users others based on similar tastes and an “attractiveness” measure on an online dating website. Hannon et al. [20] have built a people recommender specifically for Twitter. The application uses both content-based (similarity in users’ tweet content) and collaborative-based (similar followed and following users) to provide a list of recommendations to its users. Similar technology has been introduced to the Twitter platform itself28.

Many people do not use online social networks just to connect with people. Often interaction is mediated by a service or piece of content. As such, there is interest not only in how users produce and consume on these platforms, but how this information is diffused throughout a network. Blogs were an early conduit for information propagation online, and indeed research such as [18] highlights how blogs have changed the way in which people consume content online. In it, the authors track topics across a broad range of RSS channels and examine how these topics are disseminated. They find that an underlying propagation network can be found from tracking topics. Often there are individuals who are central to a network and if they have enough influence can boost a topic’s propagation throughout the network. The authors give an example of such a person: She has “a huge collection of friends, a broad set of interests, and an intelligent and up-to-date blog”. More recently, social platforms have become more ubiquitous and thus, the networks within them have matured where interaction is richer and more collaborative. Has the nature of information diffusion changed? No online social platform is more ubiquitous than Facebook, and Bakshy et al. [4] set about examining how information flows on the site. The authors find that for a user, most of their content originates from those friends they have the strongest connection with, but novel information tends to stem from the greater number of acquaintances they do not interact with as regularly.

25 http://www.linkedin.com
26 http://www.epinions.com
27 http://www.flickr.com
28 Gupta:2013
The nature and maturity of online users’ social graphs now yields a huge volume of information. Much like with the problem 1990’s web users faced, there is too much information to digest. It has become difficult to find useful socially generated content. For example, a Twitter user may have an interest in artificial intelligence but has no idea who tweet the best content related to the area. Or an Airbnb user wishes to rent a room in Barcelona but does not know which host they can trust to make good on a potential transaction. People must choose who they connect with to provide them with this useful content or service. With billions of online users collaborating on a daily basis, choosing who to collaborate with has become a major challenge. How can one user trust another if they have never previously collaborated? And how can an online user know which information is credible if they do not know who created it, or how reputable the source is? Such questions have spurred a great deal of interest in the idea of online reputation.

1.4 TOWARDS A REPUTATION WEB

The rising popularity of technologies that enable people to collaborate online shows a strong willingness among users to interact with strangers. This presents a challenge: interactions between people in the offline world naturally allow for trust building and over time, and if a person shows themselves to be trustworthy, their reputation is enhanced. Online, it is more difficult to build relationships in this way, and so if a user cannot distinguish between who is trustworthy and who is not, they may not use the system [15]. There is a growing need for these systems to provide infrastructure that enables users to foster trust, and to use this information calculate users’ reputation. The hyperlink was the key that unlocked the problem of effectively measuring the authority of web pages. With people, the problem is measuring reputation. This can be mediated by examining collaboration. Principally, online reputation can be seen as a function not just of the extent to which people collaborate, but of the quality of the collaborations they partake with others.

As discussed in this section, this information can be difficult to unpack. For example, many services encourage their users to give direct and explicit feedback on the performance of their collaborators. However that kind of information is not readily available on many other social services, and as a result quality of collaboration is not used as a primary indicator of reputation. Some real-world reputation systems look to signals such as activity levels. Others seek to enumerate more abstract concepts
like influence and reach. This section looks to past literature to define online reputation and its related concepts. It also includes an exploration of the current landscape of online reputation systems. By examining several real-world reputation systems in deployment today, it is evident that current reputation systems do not take a principled view of online reputation, rather systems tend to be ad-hoc in nature. This is a key differentiator between those systems and the approach proposed in this thesis.

1.4.1 Defining Online Reputation

Online reputation has been of great interest to both social web platforms and in research. Although the work described in this thesis discusses how reputation can be computationally calculated and applied in the online world, reputation is, at its heart, a sociological concept. A good definition of reputation is given in [17], who state that “Reputation is information used to make a value judgement about a person or a thing”. Given that this thesis discusses reputation in online collaborative platforms, we wish to focus on the reputation of people, and by association the utility of the content they create. So the question this definition begs is “how do we gain user reputation information from the system”?

One possible way is to look at how users perform in their collaborations with others. If a user has performed well in collaborations with others, their reputation is enhanced. Mui et al. in [48] were early exponents of the idea of applying a computational reputation model to foster relationship building between strangers in the online world. The authors acknowledge the relationship between trust and reputation using sociological intuition, and add a third concept to aid their definitions – reciprocity. Trust in the online world can be couched in terms of this concept, specifically that one user trusts another based on the expectation that this other user will make good on a transaction. This idea is extended to include reputation: In a community where reciprocal interaction is expected, there is a natural incentive to acquire reputation. Figure 10 illustrates the circular relationship between these three concepts. Specifically, Mui et al. state that “individuals tend to trust others with a reputation for being trustworthy and shun those deemed less so”. And so, if a person is more trustworthy in a community where reputation information is shared, it increases the likelihood of others reciprocating trustworthy behaviour. This indicates that a community which satisfies these three concepts results in a net benefit for its members.
Reputation and Related Concepts

Trust, Reputation, Reciprocity – Terms often used interchangeably. They are related, but distinct.

Trust:
A subjective expectation someone has about someone else’s future behaviour based on their past encounters.

Reputation:
Perception that someone creates through their past actions with others.

Reciprocity:
Mutual exchange of deeds (positive or negative)

net benefit

Figure 10: A diagram (introduced in [48]) illustrating the relationships between trust, reputation and reciprocity

Mui et al.’s definition builds on earlier research that focuses on offline domains such as political science [54], game theory [30] and marketplaces [55]. The latter work examines how merchant trading can take place without any kind of enforceable law protecting participants. The authors cite the medieval “Law Merchant System” as an example of a trading environment which worked based on the principal that a merchant’s past performance in transactions affected the propensity of other traders to interact with them in the future. Information on trader performance was disseminated, and judgements on disputes were made, by an impartial person known as a Law Merchant. This kind of approach relies primarily on the reputation of participants, and reputation is inferred based on the assumption that participants’ past performance is a good indicator of future performance. This approach, according to the authors was successful in ensuring the integrity of the marketplace itself, which is why it was preserved and then effectively re-applied more than 500 years later for online marketplaces.

It is quite simple to imagine how trust and reputation can be defined for a real-life merchant trading environment. Trust can be defined in terms of a relationship between two individuals – one trader trusts another if he expects beyond reasonable doubt that he will benefit from a future transaction. This trust can be based on a trader’s knowledge of the other’s reputation. As such, reputation can be regarded as a single trader’s standing in their greater community. Put simply, reputation is a function of a trader’s past collaboration performance. This performance can be measured by looking at individual trust statements that people have expressed about the trader. Despite the vast age difference between the two domains, a parallel can be drawn between the Medieval Law merchant system and online scenarios where
users interact with each other, specifically that a user’s experience is directly linked to the outcome of their collaborations with others.

Definitions for online trust and reputation are well-documented [26, 48, 23], and speak to these early sociological discussions such as the one presented in [55]. For example, Jo sang et al. [26] echo ideas put forward by Mui et al. in [48] by defining trust and reputation as distinct, but related concepts. Specifically, trust is fostered among interacting parties by knowing that each member has good reputation. In the context of an online community, the work states that “reputation can be considered as a collective measure of trustworthiness based on the referrals or ratings from members in a community”. Crucially, this work puts forward the idea that subjective trust can be derived not just from an individual’s past interaction with another member of their community, but also from past interactions the other member has had with the community at large. These core ideas have stayed more or less constant in research in online reputation (for example, see a more recent survey of reputation systems in [23]), with the changing factor being the context in which the reputation mechanism is required. For example, these systems have been employed to promote privacy [2, 13] or encourage participation [34, 36] in online social networks, to enhance robustness of online collaborative platforms like Wikipedia [11, 14], and to improve the content provided to users of recommender systems [32, 39, 52]. These domains are related to “Social Recommender Systems”, and are all discussed in this thesis.

When attempting to model user reputation on an online platform, what do those individual trust statements look like? In some cases, the platform provides functionality for its users to explicitly give feedback about other members of the community, which is an obvious proxy for trust. In other systems, this kind of mechanism is not feasible. As such, other indicators such as levels of activity or implicit feedback information are utilised. In the following sections these two forms of reputation modelling are examined in detail.

1.4.2 Feedback-based Reputation

An early domain for which reputation technology was developed was online marketplaces. The explosion in popularity of these platforms exposed them to malicious use, and as such some measure of security had to be implemented. If a user did not trust the sellers on the website, they did not trust the website itself. When a buyer and seller enter a transaction they both incur some degree of risk, for example a buyer may pay
eBay’s feedback-based reputation system. This screenshot shows the reputation of seller “unbeatable-uk”. 67,400 buyers have given them positive feedback, which accounts for 99.6% of the total feedback they have received. They average 5 stars across all performance indicators. eBay has rewarded this high-reputation user with a “top-seller” badge. This badge allows for buyers to easily find high-reputation sellers.

In the offline world, this kind of risk is mitigated by face-to-face transactions which allow for reputation to develop naturally. However in the online world, users interact remotely and often with many different partners, and so reputation is difficult to foster. eBay\(^{29}\) attempted to solve this problem with a feedback-based reputation system. In it, users can rate others depending on the outcome of transactions, and this information is publicly available to others to help them decide which users were reputable and thus safe to interact with. The core of this system is still in use today.

eBay’s reputation system allows for users to give detailed feedback about sellers, for example 5-star ratings for various performance indicators, a free-form text review and an overall positive or negative assessment. Based on this information, the reputation system assigns each seller an overall score. Figure 11 shows an example of a seller with relatively high reputation. The information presented is extremely useful to the buyer in attempting to minimize the risk they incur in making a purchase on the site. Improving system security and robustness is not the only motive behind implementing a reputation system in this context. The system also incentivises good

\(^{29}\) http://www.ebay.com
seller behaviour, as it stands to reason that users will prefer to do business with a high-reputation seller.

The type of explicit feedback mechanism employed by online marketplaces has been applied across a range of platforms and, more recently, on collaborative consumption\[6\] services where users share commodities, resources or services for a fee. Using these collaborative platforms people can share their cars on Getaround\[30\], connect with people in need of their specific skills on SkillPages\[31\], or on TaskRabbit\[32\], they can even receive payment to carry out small tasks that others in their area have outsourced. There is a similarity between these platforms and more traditional online marketplaces that allow feedback-based reputation mechanisms to work effectively: A user’s past performance in collaborations with others is a good indicator of their future performance, and thus their reputation.

A foremost website in this domain is Airbnb\[33\], a short-rentals service that offers people the chance to rent space they have, such as spare apartment bedrooms rooms or holiday properties. Although perhaps the distinction between host and renter is less well-defined than the buyer-seller relationships seen in more traditional online marketplaces, similar challenges are evident: The nature of user interaction (remote and with strangers) means that reputation-building cannot happen without help from

---

30 http://www.getaround.com
31 http://www.skillpages.com
32 http://www.taskrabbit.com
33 http://www.airbnb.com
the service itself. Airbnb takes its cues from online marketplaces and has developed a feedback-based mechanism similar to that of eBay’s. Figure 12 indicates that Airbnb’s model aggregates ratings-based and freeform text information that users provide to illustrate a user’s overall reputation. The increasing popularity of these websites indicates users’ wish to collaborate online. This is in contrast to the model of interaction on traditional online marketplaces, which has a strict dual nature. On Airbnb, a user can act as a host on one occasion, and a renter in another.

A key function that these systems serve is to encourage good behaviour and discourage bad behaviour. Many of these systems work on the basis that if a person is deemed to be reputable, they will reliably make good on transactions with other users [26]. Feedback-based systems can naturally mitigate bad behaviour, for example a low feedback score is intrinsically punitive as it discourages users from interacting with that low-score user. Looking at reputation systems in the wild, evidence suggests their success – their longevity alone showing they have at least some utility. However, research highlights the challenges that these systems face in an environment where the intentions of some users may not be genuine. In a 2002 study of eBay activity, Resnick and Zeckhauser [59] found that feedback is highly reciprocal in nature and as such, a user’s feedback score may not be a good indicator of their future performance. In worst-case scenarios, this may result in one party giving false information on another. Jøsang and Golbeck [25] claim that the success of these systems rest on their resistance to malicious use.

1.4.3 Behaviour-based Reputation

More generally, in any online collaborative system, users can assume the role of producer or consumer depending on their intention. This kind of activity is at the heart of the social web. Users collaborate not just to make profit as on Airbnb, but to share interests with friends by “liking” content on Facebook, to disseminate information by re-tweeting content on Twitter or by posting on social news sites like Reddit34, or to offer expertise on social question-and-answer (Q&A) platforms like the Stack Exchange Network35.

However, feedback-based reputation mechanisms are not always feasible on online collaborative systems, and are open to abuse. The key difference between a service

---

34 http://www.reddit.com
35 http://www.stackexchange.com
like Airbnb and a social platform like Facebook is that users do not give explicit feedback on the collaboration performance of others in their community. As such, reputation must be deduced from more implicit indicators. Although not examining reputation specifically, services such as Klout\textsuperscript{36} and PeerIndex\textsuperscript{37} examine activity-based indicators across a range of platforms like Twitter, Facebook and Linkedin\textsuperscript{38} to calculate the level of influence people have in the social media world. These sites give users a sense of how much content they produce on social media, and to what extent others consume their content. PeerIndex, for example, creates a profile for each user, showing information such as which topics they post about most frequently (see Figure 13). The primary motivation behind creating these services is to encourage deeper social media engagement. Indeed, they have added further incentives by rewarding users who have achieved a score above a certain threshold, for example reductions on items on a partner e-commerce website. Indeed, Klout will only consider prospective employees who have achieved over a score of 40 (out of a maximum 100) on their site.

\begin{thebibliography}{9}
\bibitem{36} http://www.klout.com
\bibitem{37} http://www.peerindex.com
\bibitem{38} http://www.linkedin.com
\end{thebibliography}
Some collaborative platforms have created their own bespoke mechanisms to calculate user reputation. Often these are based on activity metrics that are specific to the site. For example, the popular Stack Exchange Network, a collection of Social Q&A websites, has engineered its own approach to calculating reputation on the site, and leverages this information to encourage users to engage more in the site by publicly displaying reputation scores of community members. The system has become so popular, particularly among programmers on the network’s subsidiary site Stackoverflow\(^\text{39}\), that a member’s reputation score has become a legitimate piece of information on professional resumes. Unlike Klout and PeerIndex, a user’s Stack Exchange score is unbounded. The score is simply an aggregation of points a user has amassed by partaking in activities specific to the platform, for example, a user gains 15 points if they answer a question that the questioner indicates is correct. An added incentive to amass reputation on the site is over certain thresholds, a user gains more activity-based privileges such as the ability to edit other users’ posts.

1.4.4 Online Reputation: A Summary

The Stack Exchange’s reputation system is representative of the vast majority of systems that are employed by collaborative platforms, in that it is tailored to the specific kinds of activities users engage in on the site. If the primary activity on a platform

\(^{39}\) http://www.stackoverflow.com
is transactional in nature, like on eBay or Airbnb, then reputation can be calculated by examining explicit feedback given by its community. If users post links to content on social news sites like Reddit and Slashdot\textsuperscript{40}, then their reputation can be directly linked to how many positive votes their content has received by others. And on the Stack Exchange Network, it is down to various activity indicators specific to social Q&A. Although a bespoke system like StackExchange is ad-hoc in nature, such a system does show us that reputation is sensitive to context. For example, a person may be a great host on Airbnb, but that has no bearing on their level of expertise on StackOverflow. However, an element of uniformity is desirable. Services such as PeerIndex and Klout may give us an insight into a person’s influence on social platforms, but do they tell us much about users’ reputation? However if, as stated by [17], reputation is information used to make a value judgement about someone, it would be useful for reputation to be inferred in a similar way across collaborative platforms.

The central idea tying each of these reputation models together is collaboration. In some way, each is either explicitly or implicitly, directly or indirectly looking at information provided on user collaboration to deduce reputation. In the simplest cases, as with PeerIndex, quantity of collaboration is used. In a more sophisticated system like StackExchange, it is (partially) quality of collaboration. This thesis presents work that proposes and evaluates a principled approach to calculating online reputation that can provide uniformity while maintaining context sensitivity, but above all, reflects collaboration quality. Just as link-analysis algorithms harnessed the web of content to calculate the authority of web pages, we can now harness the web of users in a similar way. This web can be constructed by examining collaboration that occurs every day across the online world. And the collaboration graph can be harnessed to provide its users with content that is of greater quality. In the following section we describe HeyStaks, a social search utility, upon which this approach to generic reputation calculation was initially implemented and evaluated. In HeyStaks, user reputation information has been translated into item reputation to improve the quality of recommendations made to its users.

1.5 Application Case-Study: The HeyStaks Social Search Utility

This thesis details the evaluation of a computational model of reputation for online collaborative systems. For evaluative purposes, web search was used as an applica-
Figure 15: An example search session in Google. The user is searching for information on Hard Rock bikes, but is not providing Google with enough context to give the kind of results that satisfy the user. Note the top result is regarding the Hard Rock Cafe.

A reputation system was implemented for the social search utility HeyStaks, and the system was tested and evaluated in order to prove the efficacy of this approach to modelling online reputation.

Web search may seem like an unsuitable case-study for a reputation model, as web search is usually a solitary activity. Today, conventional search engines do not explicitly cater for any kind of collaboration during search. Even the simple act of sharing results between potential collaborators is not possible if solely using search platforms. For example, if I find a useful search result on information retrieval and wish to share this with a colleague, a third-party application such as a messaging tool or social web platform must be used to do so. This kind of scenario is very common, in fact, a survey conducted by Morris [45] showed that 90% of regular searchers have engaged in some sort of workaround to facilitate collaboration in search. The need
Figure 16: Again, the user searches for information on Hard Rock bikes, this time using HeyStaks. The user has joined the "Mountain Biking" stake and, as a result, the top search results relate to bikes. These top results are HeyStaks recommendations.

for collaboration runs deeper than this, for example, according to work conducted by Teevan et al. [68], 25-40% we are re-searching for content we have already found and, more pertinently, other research has shown that around two-thirds of the time we are looking for something that a friend or colleague has already found [64]. So not just is there a need for search technologies to allow for sharing of search results, but for others’ search experiences to influence those in their community.

Research in the area of collaborative information retrieval has attempted to address these issues. One such way of fostering collaboration in search is to create an interface that allows for people working in the same space, for example, the CoSearch system created by Amershi and Morris [1] allows for users to collaborate as they search on the same PC. Other similar systems use a table-top computer Smeaton et al. [62] or multiple, co-located iPod Touch devices [63]. Other systems such as $S^3$ [47]
– a system and SearchTogether [46] allow for remote, asynchronous collaboration, meaning users can integrate these systems into their workflow more easily.

These social search systems aim to integrate with an underlying search structure in order to enhance ordinary search experiences. Other systems have taken advantage of people growing accustomed to interacting online with strangers, and made the network itself the source of information for searches. Horowitz and Kamvar [24] describes a social search application that automatically routes queries to users who may be able to provide relevant information to searcher. However, this approach is limited to the availability of experts in the network.

The primary motivation behind social search technologies is to address these issues and facilitate collaboration in search. HeyStaks is a utility that aims to add collaboration features to mainstream search engines. With HeyStaks, people can use mainstream search engines like Google and Yahoo as they normally would, with the utility adding a collaborative layer on to their experience.

A key concept of the utility is the “search stak”. These can be thought of as online folders in which users can store their search experiences according to topic. Staks can be shared with other people and thus creating a mutually beneficial collaborative search environment. HeyStaks integrates with mainstream search engines by making recommendations from search staks at search time, overlaying them onto the regular search interface. 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 difficult to find if only using Google or Yahoo.

For example, imagine a web user Owen wishes to find information about the Hardrock mountain bike company41. Because of the ambiguous name of the company, Owen may have to reformulate their query several times before finding results they are happy with. Figure 15 illustrates a possible first query scenario, where Google’s top search result is the website for the Hard Rock restaurant chain – not what Owen is looking for. He eventually finds what he needs after many query attempts, and inputs his useful findings into HeyStaks, creating a search stak for this topic. Another searcher Sandra is using HeyStaks and, using the same initial query as the previous user, finds what she is looking for on the first attempt (see Figure 16). This is because she has selected the “Mountain Biking” stak to search in. In this case, Owen is pro-

HeyStaks captures the activities of its users via a web browser toolbar. This toolbar can record users’ search result selections, but also provides users with features to express more explicit interest in a page, such as tagging a result with text, vote up or down a result, or explicitly share a particular result with another community member. A drop-down menu allows users to select which stak to search within. Users can also create, share and join staks via the toolbar. Staks can be set to “public” or “private” (invitation-only), allowing users a degree of control over who can access the search knowledge staks contain. Recommendations stem primarily from a the stak the user has currently selected, though recommendations can also be made from other staks the searcher is a member of.

The client-side toolbar is one of two key components that make up the HeyStaks utility. The other is a back-end server, which manages 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 the recommendation engine. Briefly, the recommendation engine creates a candidate list of result pages by looking at term data and usage data. Term data is gained by matching query terms to terms in the stak index and then scoring each page using a boosted \textit{TFIDF} retrieval function\footnote{\url{http://lucene.apache.org/core/old_versioned_docsversions/2_9_0/api/org/apache/lucene/search/Similarity.html}}. For each candidate page, another score is calculated based on the positive user actions on that page (result selections, tags, positive votes, shares etc). These two scores are weighted against each other to achieve an overall relevance score, and a ranked list of recommendations is generated. The recommendation engine is discussed in more detail in \cite{40} and \cite{65}.

Figure 18 illustrates this architecture, with the addition of a reputation engine as a key component. The engine effectively alters HeyStaks’ recommendation process by adding weighting to recommendation candidates according to the reputation of their sources. Each page’s reputation score then, has some effect on its position in the ranked list of recommendation candidates. As such, not only can the score change the position of a result on the list returned to the user, but whether the page is returned at all. Integrating the reputation system in this way allows for a clear method of eval-

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure17.png}
\caption{The HeyStaks browser toolbar.}
\end{figure}
Figure 18: HeyStaks’ back-end server architecture. The reputation engine is fully integrated, generating scores for recommendation candidates based on that of the users who produced them.

Evaluation: Does reputation information improve the relevance of results presented to users over time, against a recommendation engine that does not consider reputation? Before answering this question, the issue of modelling user reputation in HeyStaks must be broached, as it is user reputation that is used as a source for information in this new HeyStaks architecture. As indicated above, user reputation can be deduced by examining the collaborative behaviour of HeyStaks users. And so, this case-study serves to answer two key research questions posed by this thesis:

1. Is it possible to calculate user reputation by looking at how they collaborate with others in their community?

2. Can user reputation be harnessed to calculate the reputation of the content they produce, and indeed, positively influence the experience of those that consume that content?

In HeyStaks, collaboration occurs naturally between searchers. By simply acting on a HeyStaks recommendation, a user is collaborating with the producer(s) of that rec-
From User to Result Reputation

Figure 19: Reputation of a page in HeyStaks can be calculated by translating the reputation of its producers. Note that due to the type of activity seen on the platform, all users who have acted on a page can be thought of as the producers of that page. In this case, recommendation (see Figure 19. As all collaborative activity is recorded by HeyStaks, a reputation module can be implemented and directly integrated into the system’s server architecture.

1.6 CORE CONTRIBUTIONS

1.6.1 A Computational Model of Reputation

The primary contribution of this thesis is the presentation of a reputation model based on an underlying network of collaborations that occur on social web platforms. The model itself has three constituent parts:

- The Fundamental Units of Collaboration We describe how interactions (both implicit and explicit, and direct or indirect interactions) between producers and consumers constitute distinct collaboration events. These trigger the flow of units of trust from the consumer to the producer.

- A Graph-based Model of Collaboration We can use the information gained from recording individual collaboration events to link collaborative users together as a collaboration graph, with the nodes representing producers and consumers,
and the links individual instances of collaboration. This graph makes explicit the interactions between users adopting distinct roles – producers and consumers – with respect to the sharing and consumption of information in the network.

- *Modeling User Reputation* We describe how reputation can accumulate over time. Each node in the collaboration graph is examined, and trust scores that have been conferred to users according to the principal of the collaboration event can be aggregated to measure individual users’ reputation in their online community.

1.6.2 *From Reputation to Recommendation*

We go on to show how this model of reputation can be used as part of a recommender system.

- First, we describe how reputation can be transferred from users to items. The reputation of an item can be realised by examining that of one or more of its producing users. We explore a number of different methods for carrying out this translation, with the aim of maximising the model’s effectiveness in a recommender system environment.

- Secondly, we show how item reputation can be used to guide recommendation. We apply this concept to a social recommender system, which allows us to measure the utility of our approach. We discuss our different approaches to calculating both user and item reputation in the context of their performance in delivering relevant recommendations to users.

1.6.3 *Generalising the Approach*

We illustrate how our approach can be generalised for use in two task domains. The primary domain is the social search utility HeyStaks, which recommends search content to its users. Secondly, we demonstrate preliminary work in the area of Social Question and Answering systems.
Part I

A COMPUTATIONAL MODEL OF REPUTATION

The key idea presented in this thesis is that reputation can be inferred by examining how users collaborate on online social systems. Our objective is to model and harness reputation with a view to building a better recommendation system. As a starting point, in this section, we describe our work on modeling collaboration and reputation in online systems, focusing on social search via the HeyStaks utility.

In particular we present two papers. In the first paper, we introduce our collaboration-based reputation model. It describes the collaboration event, and introduces early ideas about how reputation can be modeled and measured using a graph of events called the collaboration graph. For supplementary information on the experimental methodology, see A.1.

In the second paper we describe the results of a detailed evaluation of this reputation model in HeyStaks, using live-user data. The main focus of this paper is to explore the dynamics of the reputation model with a particular emphasis on the relationship between reputation and search expertise. The paper also introduces the idea that user reputation can be used to influence recommendations, which motivates the next section of the thesis. Supplementary information on the set-up of this experiment can be found in A.2.
PAPER 1: TOWARDS A REPUTATION-BASED MODEL OF SOCIAL WEB SEARCH

Kevin McNally, Michael P. O’Mahony, Barry Smyth, Maurice Coyle and Peter Briggs
Published in IUI ’10: Proceedings of the 15th International Conference on Intelligent User Interfaces (2010), pp 179–188
Towards a Reputation-based Model of Social Web Search

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

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

Author Keywords
Collaborative Web Search, Reputation Model, HeyStaks

ACM Classification Keywords
H.4.0 Information Systems Applications: General

General Terms
Algorithms, Experimentation, Security

INTRODUCTION
The world of web search is usually viewed as a solitary place. Although millions of searchers use services like Google and Yahoo everyday, their individual searches take place in isolation. Recently, researchers have begun to question the solitary nature of web search, proposing a more collaborative search model in which groups of users can cooperate to search more effectively [10, 11, 12, 13, 17]. Indeed, recent work by [4] highlights the inherently collaborative nature of more general purpose web search. Despite the absence of explicit collaboration features from mainstream search engines, there is clear evidence that users implicitly engage in many different forms of collaboration as they search – although these collaboration “work-aro~s” are often frustrating and inefficient [4]. Naturally, this has motivated researchers to consider how different types of collaboration might be supported by future editions of search engines. HeyStaks is one such model of collaborative web search (www.heystaks.com) which has been designed to work with mainstream search engines, such as Google, and which has recently been deployed online. HeyStaks takes the form of a browser toolbar to allow users to capture and share their search experiences with other users and, in so doing, facilitates the creation of search communities. In turn, members of these search communities benefit from recommendations that are derived from the activities of other community members.

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

COLLABORATIVE INFORMATION RETRIEVAL
Collaborative information retrieval research takes a fresh look at information retrieval and web search, which highlights the potential for collaboration between searchers during extended search tasks. Recent work by [4] highlights the inherently collaborative nature of more general purpose web search. For example, during a survey of just over 200 respondents, clear evidence for collaborative search behaviour emerged. More than 90% of respondents indicated that they frequently engaged in collaboration at the level of the search process. For example, 87% of respondents exhibited “back-seat searching” behaviours, where they watched over the shoulder of the another searcher to suggest alternative queries. Some 30% of respondents engaged in search coordination activities, by using instant messaging to coordinate searches. Fur-
thermore, 96% of users exhibited collaboration at the level of search products, that is, the results of searches. For example, 86% of respondents shared the results they had found during searches with others by email. Indeed, almost 50% of respondents telephoned colleagues directly to share web search results, while others prepared summary documents and/or web pages in order to share results with others.

Thus, despite the absence of explicit collaboration features from mainstream search engines, there is clear evidence that users implicitly engage in many different forms of collaboration as they search, although, as reported by [4], these collaboration “work-arounds” are often frustrating and inefficient. This has motivated researchers to consider how different types of collaboration might be supported by future editions of search engines. The resulting approaches to collaborative information retrieval can be distinguished in terms of two dimensions – time and place. In terms of the former, collaborative search systems can be designed to support synchronous or asynchronous collaborative search. And in terms of the latter, systems can be designed to support either co-located or remote forms of collaborative search.

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

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

**HEYSTAKS: A SEARCH UTILITY**

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

**System Architecture**

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

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

**A Worked Example**

To make HeyStaks more concrete, it is useful to consider a worked example. With this in mind, consider the scenario where the leader of a recommender systems research group wishes to harness the search knowledge of his/her group to help other group members, particularly new researchers, to search more productively.
This is the scenario illustrated in Figure 2. To begin with, the group leader creates a new search stak by selecting the “Create a New Stak” option from the “Staks” menu in the HeyStaks toolbar. As per Figure 2(a), creating a stak is a straightforward process: the stak creator needs to provide a stak name and some helpful description information; the stak can be configured to be public (anyone can join) or private (invitation only); and the creator can invite initial members by providing their email addresses. In this case the user creates the public RecSys stak and invites a group of researchers via the postgrads@clarity-centre.org group-email address. If the researchers accept this invitation, then the RecSys stak will be added to their HeyStaks toolbar.

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

In this way, as stak members submit queries and select results, these search experiences are captured in the RecSys stak. As mentioned above, HeyStaks also allows users to more explicitly interact with search results and web pages. For example: users can vote for (or against) particular results; users can email a page directly to another user without leaving the page or their search; and users can explicitly tag any page that they find interesting (see Figure 2(c)). This combination of implicit click-thru data and explicit voting, sharing, or tagging data permits staks to capture a variety of important interaction types, which HeyStaks uses to infer the relevance of a page to a given stak; see [19] for details.

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

### The HeyStaks Recommendation Engine

In HeyStaks, each search stak \( S \) serves as a profile of the search activities of the stak members. Each stak is made up of a set of result pages \( S = \{p_1, ..., p_k\} \) and each page is anonymously associated with a number of implicit and explicit interest indicators, including the total number of times a result has been selected \((sel)\), the query terms \( q_1, ..., q_n \) 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 Eq. 1.

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

In this way, each page is associated with a set of term data (query terms and/or tag terms) and a set of usage data (the selection, tag, share, and voting count). The term data is represented as a Lucene (lucene.apache.org) index, with each page indexed under its associated query and tag terms, and provides the basis for retrieving and ranking promotion candidates. The usage data provides an additional source of evidence that can be used to filter results and to generate a final set of recommendations. At search time, recommendations are produced in a number of stages: first, relevant results are retrieved and ranked from the stak index; next, these promotion candidates are filtered based on the usage evidence to eliminate noisy recommendations; and, finally, the remaining results are added to the Google result-list according to a set of recommendation rules.

### Retrieval & Ranking

Briefly, there are two types of promotion candidates: primary promotions are results that come from the active stak \( S' \); whereas secondary promotions come from other staks in the searcher’s stak-list. To generate these promotion candidates, the HeyStaks server uses the current query, \( q_t \), as a probe into each stak index, \( S_i \), to identify a set of relevant stak pages, \( P_i(S_i, q_t) \).

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

\[
score(q_t, p) = \sum_{teq} tf(t)p \cdot idf(t)^2 \quad (2)
\]

### Evidence-Based Filtering

Staks are inevitably noisy, in the sense that they will frequently contain pages that are not on topic. As a result, the retrieval and ranking stage may select pages that are not strictly relevant to the current query context. To avoid making spurious recommendations, HeyStaks employs an evidence filter. This filter uses a variety of threshold models to evaluate the relevance of a particular result in terms of its usage evidence. For example,
a) The user creates a RecSys stak as a repository for searches related to recommender systems work. Sharing the stak with research students allows others to contribute to, and benefit from, its growing search knowledge.

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

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

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

e) Users can search for new staks to join, given an information need. Here we see how a search for "collaborative filtering" has revealed a number of relevant staks, which the user can join or share with others.

Figure 2. HeyStaks in action: a) stak creation; b) result recommendations; c) tagging a web page; d) stak activity on the HeyStaks portal; e) finding new staks.
tagging evidence is considered more important than voting, which in turn is more important than implicit selection evidence. Pages that have only been selected once, by a single stak member, are not automatically considered for recommendation by HeyStaks and, all other things being equal, will be filtered out at this stage. In turn, pages that have received a high proportion of negative votes will also be eliminated. The precise details of this model are beyond the scope of this paper but suffice it to say that any results which do not meet the necessary evidence thresholds are eliminated from further consideration.

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

Summary Discussion
HeyStaks is designed to help users to collaborate during web search tasks and, importantly, it succeeds in integrating collaborative recommendation techniques with mainstream search engines. In preceding sections, we have provided an overview of the various functionality that HeyStaks provides and have discussed the ranking, filtering and recommendation techniques that are used to make result promotions. Further details on the precise techniques employed can be found in previous research [2, 18]. The HeyStaks system has recently moved into public-beta and during this time approximately 500 users have registered, leading to the creation and sharing of thousands of search staks. In the next section, we introduce a reputation model for HeyStaks and we show how this model can be used to identify the search leaders in a community and also how user reputation can be used to enhance the ranking of result promotions.

A REPUTATION MODEL FOR SOCIAL SEARCH
As described in the previous section, the many and varied different types of activities that a user can perform (click-thru, tagging, voting, sharing) on a web page are ultimately combined and leveraged by HeyStaks to make recommendations at search time. And, while the recommendation algorithm used differentially weights different activity types (so that tagging, for example, is considered a more reliably indicator of interest 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 and, as such, perhaps their activities should be considered as more reliable at recommendation time, so that promotion candidates that hail from the activities of very experienced users might be considered ahead of candidates that come from the activity of less experienced users. This is particularly important given the potential for malicious users to disrupt stak quality by introducing dubious results to a stak.

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

From Activities to Reputation
It seems natural that the reputation of a 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 according to a variety of different types of user activities, which include:

• Click-thrus (Result Selections) – that is, a user selects a search result (whether organic or promoted);

• Voting – that is, a user positively votes on a given search result or the current web page;

• Sharing – that is, a user chooses to share a specific search result or web page with another user (via email or by posting to their Facebook Wall etc.);

• Tagging/Commenting – that is, the user chooses to tag and/or comment on a particular result or web page.

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

What then is the relationship between search activity and searcher reputation? Under the heading of “more search knowledge is better than less search knowledge” it might make sense to model reputation as a direct function of the sheer volume of activity that a given searcher engages in. This would be a mistake. For a start, just because a user is creating a lot of search knowledge, by adding many pages
to search staks, it does not mean that this new knowledge is useful, especially to others. On the contrary, one of the major concerns in any social recommender is the potential for misuse through the actions of malicious users, a problem that would no doubt be exacerbated by valuing the contribution of very ‘productive’ malicious users.

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

Reputation as Collaboration

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

If HeyStaks recommends a result to a searcher, and the searcher chooses to act on (select, tag, vote on or share) this result, then we can view this as a single instance of search collaboration. The current searcher who chooses to act on the recommendation is known as the consumer and, in the simplest case, the original searcher whose earlier action on this result caused it to be added to the search stak is known as the producer. In other words, the producer created search knowledge that was deemed to be useful enough for the consumer to act upon it. And the basic idea behind our reputation model is that this act of implicit collaboration between producer and consumer confers a unit of reputation on the producer (Figure 3). If a given user is a regular producer of search knowledge that is frequently recommended to, and acted on by, many other users, then this producer will enjoy a high reputation score. Moreover, if users create lots of staks and share these staks with many other users, or simply join staks that have been created by others, then they create an opportunity for more collaboration; and if users contribute good search knowledge to shared staks then their reputation score will benefit from the realisation of these frequent collaboration opportunities. In this way, this collaboration-based model of reputation is incentivizing users not just to create search knowledge but also to share it with others.

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

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

An Example

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

$t_1$: User $u_1$ organically selects page $p$ for some search query $q$, causing page $p$ to be added to stak $S$.

t_2$: User $u_2$ selects page $p$, which has been promoted by HeyStaks, for a search query that is related to $q$. Since user $u_1$ is the only user to have previously selected page $p$ in stak $S$, we say that user $u_1$ (the producer) has promoted page $p$ to user $u_2$ (the consumer). Consequently, user $u_2$ assigns a reputation score of 1 to user $u_1$.

t_3$: User $u_3$ organically selects page $p$ for an unrelated search query $q’$. This time, page $p$ is not promoted by HeyStaks and hence no reputation is assigned by user $u_3$ to any of the other users.

t_4$: Finally, user $u_4$ selects page $p$, which has been promoted by HeyStaks, for a search query that is again related to $q$. Since users $u_1, u_2$ and $u_3$ have all previously selected (either organically or by promotion) page $p$, on this occasion reputation is assigned by user $u_4$ to each of these users. Thus, in Figure 4, the reputation score is distributed equally among the three users, such that each user receives a score of $1/3$. 

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

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

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

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

**Result Promotion**

We now consider how user reputation can be employed to influence the ranking of promoted results in HeyStaks. Currently, pages are selected for promotion as follows. For a given search query $q_t$ submitted by a user $u$, a set of candidate pages, $\{p_1, p_2, \ldots, p_k\}$, are identified for promotion and a relevance score, $\text{score}(q_t, p_i)$, is computed for each page (Eqn. 2). These scores are then used to rank order candidate pages, and the pages with the highest scores are promoted to the user.

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

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

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

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

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

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

**PRELIMINARY EVALUATION**

Ultimately it is our intent to evaluate this reputation model as an integrated component within HeyStaks as part of a long-term user trial. Accordingly it will be possible to determine just how important a role reputation can play when it comes to influencing recommendations and user engagement. For example, will a reputation-based recommenda-
tion model lead to improved recommendations that attract more frequent click-thrus? And would exposing reputation statistics to users help to deepen their engagement with the system, leading to more collaboration in the long-run?

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

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

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

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

![Graph: User search activity versus count](Image)

Figure 6. User search activity versus count

Figure 7(a) shows summary statistics in relation to the total activity (i.e. selection, tagging, voting, sharing) carried out by the 26 users. The median total activity across the users is 638 with the most active user performing as many as 3528 activities, while the least active user performed only 11 actions. This suggests a reasonable spread of activity across the 26 test users.

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

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

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

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

The results in Figure 8 tell a tale of two user types. On the one hand, about 20% of the users (5 out of 26) have achieved reputation scores of 60 or higher, with one user achieving a reputation score in excess of 105. These are clearly users who are engaged in a significant amount of search collaboration as distinct from the other users (21 out of 26) who have
helped to drive some, although a relatively small amount of, collaboration. These users have reputation scores between 1 and 25 and as such have proven to be less important when it comes to producing search knowledge that is of use to others (although no doubt these users are consuming promotions that others have produced). We view the smaller subset of high-reputation users as the search leaders within HeyStaks, while there is a larger subset of search followers who are more likely to consume than produce search knowledge.

CREATING AND JOINING STAKS MAY BE NECESSARY BUT IT IS NOT SUFFICIENT TO DRIVE REPUTATION, AT LEAST IN THE ABSENCE OF THE CONTRIBUTION OF THE USER, THROUGH THEIR ACTIVITY, TO THEIR SEARCH NETWORK’S KNOWLEDGE. CORRELATION WITH NETWORK SIZE ACROSS USERS IS COMPARATIVELY LOW, SUGGESTING THAT WHILE HAVING A LARGE NETWORK DOES INDEED PROVIDE A USER WITH A GOOD OPPORTUNITY TO COLLABORATE, SOME USERS MAY PREFER TO WORK TOGETHER IN MORE TIGHTLY-KNIT SEARCH COMMUNITIES.

Figure 8. Reputation scores for 26 users

**Activity versus Reputation**

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

Nonetheless, it is interesting to understand the strength of the relationship between factors such as activity, staks created/joined, network size and user reputation scores. This correlation information is presented in Figure 9 and it is interesting to note that there is a clear relationship between the factors and the user reputation score. Overall, the degree of user activity comes out on top compared to the number of staks created or joined and network size. This shows that creating and joining staks may be necessary but it is not sufficient to drive reputation, at least in the absence of the contribution of the user, through their activity, to their search network’s knowledge. Correlation with network size across users is comparatively low, suggesting that while having a large network does indeed provide a user with a good opportunity to collaborate, some users may prefer to work together in more tightly-knit search communities.

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

**CONCLUSIONS**

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

The main contribution of this paper, however, is the proposal of a reputation system for HeyStaks as a way to model the value of individual searchers, in terms of a reputation score, in order to weight their contributions during result recommendation. The key insight behind the proposed model is that the reputation of a user can be best measured by looking at how often the user is responsible for result recommendations that are ultimately selected. We have described how reputation can be calculated directly from user activity and we have provided some preliminary results based on an analysis of user activity and reputation scores across a sample of HeyStaks users. In summary the reputation model, while correlated with factors such as the number of staks
created/joined, the size of a user's network, and the activity level of the user, is not dominated by any single factor and so should help to preserve the integrity of the model.

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

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

ACKNOWLEDGEMENTS
Based on works supported by Science Foundation Ireland, Grant No. 07/CE/11147 and HeyStaks Technologies Ltd.

REFERENCES
PAPER 2: A CASE-STUDY OF COLLABORATION AND REPUTATION IN SOCIAL WEB SEARCH

Kevin McNally, Michael P. O’Mahony, Barry Smyth, Maurice Coyle and Peter Briggs
Published in ACM TIST: Transactions on Intelligent Systems Technology 3.1 (2011): 4
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

Although collaborative searching is not supported by mainstream search engines, recent research has highlighted the inherently collaborative nature of many web search tasks. In this paper, we describe HeyStaks (www.heystaks.com), a collaborative web search framework that is designed to complement mainstream search engines. At search time, HeyStaks learns from the search activities of other users and leverages this information to generate recommendations based on results that others have found relevant for similar searches. The key contribution of this paper is to extend the HeyStaks social search model by considering the search expertise, or reputation, of HeyStaks users and using this information to enhance the result recommendation process. In particular, we propose a reputation model for HeyStaks users that utilises the implicit collaboration events that take place between users as recommendations are made and selected. We describe a live-user trial of HeyStaks that demonstrates the relevance of its core recommendations and the ability of the reputation model to further improve recommendation quality. Our findings indicate that incorporating reputation into the recommendation process further improves the relevance of HeyStaks recommendations by up to 40%.

Categories and Subject Descriptors: H.4.0 [Information Systems Applications]: General
General Terms: Algorithms, Experimentation, Security
Additional Key Words and Phrases: Trust, Reputation, Social Search, HeyStaks

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; Spertetta 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.

\(^1\)http://www.boygeniusreport.com/2010/07/07/twitter-handling-24-billion-search-queries-per-month/

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.

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.

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 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-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.
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.

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 \((S = \{r_1, \ldots, r_k\})\) 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 irrelevant to their current needs. Nevertheless, the frequent selection of a specific page in a specific stack, 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 choose 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 stack \( S \) 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.

$$\begin{align*}
  r_i^S &= \{ q_1, ..., q_n, t_1, ..., t_m, v^+, v^-, sel, tag, share \} .
\end{align*}$$

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 stack 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 stack \( S_i \); whereas secondary recommendations come from other stacks in the searcher’s stack-list. To generate these recommendation candidates, the HeyStaks server uses the current query \( q_t \) as a probe into each stack index, \( S_i \), to identify a set of relevant stack 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.

$$\text{realscore}(q, r) = \sum_{t \in q} tf(t) \times idf(t)^2.$$ (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
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 ($\text{rep}$) 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 $\text{rep}/k$ units of reputation.

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.

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
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 $0$.

There 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.

$$producers(r, t) = \{p_1, ..., p_k\}.$$  \hspace{1cm} (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.

$$rep(p_i, t) = rep(p_i, t - 1) + 1/k.$$  \hspace{1cm} (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
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 relevence 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 fol-
lowing 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 repu-
tation 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 $n_i(p_i, t - 1)$ 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 $n_i(p_i, t - 1)$ 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; $\kappa$ 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 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.

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

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.
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.

$$\text{repscore}(r, t) = \max_{\forall p_i \in \{p_1, \ldots, p_k\}} \left( \text{rep}(p_i, t) \right).$$

(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 $p_1, \ldots, p_k$ 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 $w$.

$$rankscore(r, q_t, p_1, \ldots, p_k, t) = w \times repscore(r, t) + (1 - w) \times relscore(q_t, r).$$  \hfill (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 stack 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 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
Table I. The questions presented to trial participants. The correct answers for each question are also shown.

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

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.

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

lected. 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 stacks, 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 stacks 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?

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 activities for the members of the shared staks; this is a relative increase of 22% – 74% in favour of the shared staks.

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.

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.

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.

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.

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.

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.
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.

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.

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.

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 \( \geq 0 \)), 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 threshold) 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 stacks on topics of their choosing. At search time, HeyStaks learns from the search activities of the members of a stack 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 stacks 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 stack, 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


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.


A Case Study of Collaboration and Reputation in Social Web Search


Part II

FROM USER REPUTATION TO ITEM REPUTATION FOR RECOMMENDATION

The findings in the first two papers showed that it is indeed possible to calculate user reputation in an online social platform based on collaborations that occur naturally between its users. This is done by first awarding reputation to users as individual interactions happen, and then aggregating reputation at the community level. Such a way of calculating reputation was defined by key past research [23, 26, 48], as discussed in Section 1.4.1. Findings in early work presented here also indicated that item reputation, translated from user reputation, can be used to positively influence a social recommender system.

This part of the thesis contains three papers that describes and evaluates a number of different approaches to calculating reputation. It also considers how reputation can be aggregated to optimally improve recommendation relevance by calculating user and item reputation by comparing a number of different scoring techniques.

The first paper in this section describes how ongoing collaboration, recorded as events, can be viewed as a collaboration graph, where the nodes are users and the arcs are individual collaborations between them. This graph can be analysed in order to infer user reputation. A number of techniques to score users are tested in terms of their effectiveness at improving recommendation quality. In the second paper a similar analysis is carried out, this time exploring different methods for translating user to item reputation.

Finally, the third paper explores all possible combinations of user and item reputation scoring metrics in order to find the best possible approach to calculating reputation with the goal of improving recommendation quality.
PAPER 3: EVALUATING USER REPUTATION IN SOCIAL WEB SEARCH

Kevin McNally, Michael P. O’Mahony, Barry Smyth
Published in RSWeb ’11: 3rd Workshop on Recommender Systems and the Social Web, in association with the 5th ACM Conference on Recommender Systems (2011)
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 Josang 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 Schifanella 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, $S = \{r_1, ..., r_k\}$. 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 $r_i^S$ from stak $S$, is associated with relevance indicators: the number of times a result has been selected ($Sl$), the query terms ($q_1, ..., q_n$) that led to its selection, the terms contained in the snippet of the selected result ($s_1, ..., s_m$), the number of times a result has been tagged ($Tg$), 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.

$$r_i^S = \{q_1, ..., q_n, s_1, ..., s_m, t_1, ..., t_m, v^+, v^-, Sl, Tg, Sh\} \quad (1)$$

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 candidates 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.

$$\text{score}(r, q) = w \times \text{rep}(r, t) + (1 - w) \times \text{rel}(q, r).$$  \hspace{1cm} (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.

$$\text{rel}(q, r) = \sum_{t \in q} \text{tf}(t, r) \times \text{idf}(t)^2.$$  \hspace{1cm} (3)

The reputation of a result $r$ at time $t$ ($\text{rep}(r, t)$) is an orthogonal measure of recommendation quality. The intuition is that we should prefer results that originate from more reputable 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 candidates originating from the activities of very experienced users, for example, 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. 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 [3, 7, 13, 15]. For this reason we propose to complement the relevance of a page, during recommendation, with an orthogonal measure of reputation 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.

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 Collaboration Events

Our first and simplest reputation model calculates the reputation 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 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. 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 \( n_r(p_i, t-1) \) be the number of these results that have been subsequently recommended and consumed by other users. We define the consumption ratio according to Equation 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.

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

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.

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

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:

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

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 set of outlinks from \( p_j \) (i.e. the other users from whom \( p_i \) 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 \( L(p_i) \) is the set of outlinks from \( p_i \) (i.e. the other users from whom \( p_i \) 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.

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

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 scores 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 \( qr \).

5. EVALUATION

In previous work [19] we have demonstrated how the standard relevance-based recommendation techniques used 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 recommendations, 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 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>
<thead>
<tr>
<th>Question</th>
<th>Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. Which Old Testament book is about the sufferings of one man?</td>
<td>John</td>
</tr>
<tr>
<td>3. Which reporter fronted the film footage that sparked off Band Aid?</td>
<td>Andrew Bashir</td>
</tr>
<tr>
<td>4. Which space probes failed to find life on Mars?</td>
<td>Galileo, Mars Express, Cassini, Juno, New Horizons, Voyager 1, Voyager 2</td>
</tr>
</tbody>
</table>

Table 1: A sample of the user-trial questions.
ations 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 recommended results that are relevant to the percentage where the top result was deemed to be relevant, to the percentage of those where the top result was not-relevant. Moreover, we can compare this to the relevance rate of the 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>0.45</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.
trial to compute an overall relevance benefit: such that a relevance benefit of 40%, for a given reputation model, means that this model generated 40% more relevant recommendations than the standard HeyStaks ranking scheme.

Figure 3 presents a graph of relevance benefit versus the weighting ($w$) used in Equation 2 to adjust the influence of term-based relevance versus user reputation during recommendation. The results for all four reputation models indicate a significant benefit in recommendation quality when compared to the standard HeyStaks recommendations. As we increase the influence of reputation over relevance during recommendation (by increasing $w$) we see a consistent increase 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 reputation models are driving a relative improvement in recommendation 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 maximum 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 context. 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 manner 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 producers (i.e. producers with high Authority scores) — serve 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 reputation 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 themselves 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 searches to generate reputation-enhanced recommendations. In this evaluation we have compared 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 recommendations 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 genuine 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 ranking signal. Of course this evaluation cannot tell whether users will actually select these reputation ranked recommendations, although there is no reason to suppose that they would treat these recommendation differently from the default 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


PAPER 4: MODELS OF PAGE REPUTATION IN SOCIAL SEARCH

Kevin McNally, Michael P. O’Mahony and Barry Smyth
Models of Web Page Reputation in Social Search

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

Abstract—To date web search has been a solitary experience for the end-user, despite the fact that recent studies highlight the potential for collaboration that is inherent in many search tasks and scenarios. As a result, researchers have begun to explore the potential for a more collaborative approach to web search, one in which the search actions of other users can influence the results returned. In this context, the expertise of other users plays an important role when it comes to ensuring the quality of recommendations that arise from their actions. The reputation of these users is important in collaborative and social search tasks, much as relevance is vital in conventional web search. In this paper we examine this concept of reputation in collaborative and social search contexts. We describe a number of different reputation models and evaluate them in the context of a particular social search service. Our results highlight the potential for reputation to improve the quality of recommendations that arise from the activities of other searchers.

I. INTRODUCTION

There is little doubt the impact that search engines have had on the internet and on the lives of internet users. Most of us routinely turn to our favourite search engine when we want to locate information online. The success of web search has come about as a result of some very significant innovation that dates back to the early days of web search in the late 1990s. At the time, the first generation of web search engines (e.g. Altavista, Excite, Lycos etc.) relied on an information retrieval type approach to search, selecting and ranking results based largely on how well they matched the terms in the current search query. It quickly became apparent, however, that such an approach would not scale in the world of the web. While the presence of query terms in a result page might signal potential relevance, this type of matching did not provide a strong enough ranking signal and consequently search engine result lists performed poorly in terms of their overall precision. The significant breakthrough that led to modern web search engines came about through the work of Brin and Page [1], and Kleinberg [2], 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.

Today’s search engines still rely on link connectivity, term overlaps, and other relevance signals. In the meantime however, the so-called social web has provided users with new types of information discovery tools. Many users frequently find that the content they are interested in is shared via their social graph; for example, page sharing on Twitter and Facebook is now commonplace and many users rely on their social feeds as a source of daily content. Consequently, there is now considerable interest in the concept of social search. If our social graphs are a valuable source of content then why not harness our social networks to improve mainstream search results? As a result, we have seen major search engines like Bing and Google introduce Twitter and Facebook feeds into the result pages and both search engines now provide users with an opportunity to effectively vote on pages (through Facebook “likes” and Google +1) so that this social signal can be used during result ranking. In this paper we focus on the HeyStaks system [3]. HeyStaks 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 searches of people we trust on topics that matter to us. Previous work has focused on the HeyStaks recommendation engine and on a novel model of user reputation based on instances of collaboration between users [4].

In this paper we describe in detail how user reputation can be used during the result recommendation process. While earlier work evaluates different types of user reputation models, in this work we describe how reputation from multiple users can be combined and aggregated to model page reputation, which can provide evidence for result quality and relevance at recommendation time. Specifically, we describe and evaluate 5 different reputation aggregation techniques and evaluate their impact on the quality of recommendations based on the results of a live-user trial. The remainder of this paper is structured as follows. In the next section we outline related work in the area of web search and recommender systems. In sections III and IV we provide a brief overview of the HeyStaks system and its user reputation model, which will serve as the basis for this work. In section V we describe a number of different page reputation models and show how they can be used to influence recommendations at search time. Finally, before concluding, in section VI we describe the results of a comparative study of these strategies, based on live-user data.

II. RELATED WORK

Recently, research efforts have been focused on identifying new signals that can be harnessed to assist users to locate
relevant search results and product and service recommendations on the web. Here we briefly review some related work in this regard, examining some of the relevance models that drive modern web search engines and recommender systems. Further we describe recent work in the area of reputation systems and how reputation can be used to further enhance the quality of search results and recommendations that are delivered to end users.

One of the key differentiators between web search engines and traditional information retrieval approaches is that the underlying link structure of the web can be leveraged by search engines as a additional source of relevance and ranking information to complement query term matching techniques. Two of the best known examples of web search algorithms that utilise link structure are Google’s PageRank algorithm [1] and the HITS algorithm developed by Kleinberg [2]. The PageRank algorithm models pages on the web as vertices in a directed graph with the hyperlinks between pages representing the edge set. The relative importance of a page is modeled by the number of inlinks to the page, which can be seen as a kind of recommendation from the wider community. PageRank is a recursive algorithm, where the ranks of pages are a function of the ranks of those pages that link to them, with pages that are linked to by many other important pages receiving higher ranks themselves. The HITS algorithm also utilizes link structure to rank web pages. In contrast to PageRank, HITS computes an authority and a hub score for each page, which measure the value of a page’s content and the value of its links to other pages, respectively. There is no doubting the value and success of mining the link structure of the web as a page ranking signal; a recent comScore press release indicates that Google Sites achieved 65.4% market share in the U.S. explicit core search market.

In addition to conventional query-based search services, recommender systems have also been widely employed on the web to help people discover information, products and services that are relevant to their personal needs. In the literature, two main approaches to recommender systems have been described, known as content-based [5] and collaborative-filtering based [6] approaches. In content-based systems, items are recommended to users that are similar to items that have been liked in the past. Comparisons between items are calculated over the features that are associated with each item (for example, movies can be described by meta-data such as genre, director, cast etc.). Content-based approaches have been used in a variety of recommendation applications, including TV, e-commerce and travel recommenders [7], [8]. On the other hand, collaborative recommenders help users to make choices based on the preferences of other users in a system. The basic heuristic employed is that users who agreed or disagreed on items in the past are likely to agree or disagree on future items. To make recommendations, collaborative filtering algorithms typically find the most like-minded users in a system based on the similarity of their preference data, and weight and combine the preferences of those users. A key advantage of collaborative recommenders is their ability to capture relationships between users based on item preference information alone, and without the need for any rich meta-data about the item being recommended. In addition, collaborative and content-based techniques can be combined to form hybrid recommenders as described in [9].

Recently there has been considerable interest in reputation systems as an additional signal to enhance the quality and robustness of recommendations made to users. The work of O’Donovan and Smyth [10] addresses reputation in the context of collaborative 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%. Similar to O’Donovan and Smyth, Massa and Avesani [11] 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. The authors find that MoleTrust can improve the accuracy of predictions made by a recommender system, even in cases where users have provided few item ratings.

In this paper we show how reputation can be leveraged to influence the ranking of result page recommendations made by the HeyStaks social search service. In particular, we propose a number of models to estimate page reputation based on the reputation of those HeyStaks users who have previously interacted with the page. As with the related work discussed above, our findings indicate that the signal afforded by our reputation model leads to enhanced recommendation quality when combined with the HeyStaks page relevance model as described in the next section.

III. A REVIEW OF HEYSTAKS

HeyStaks is an approach to collaborative web search that is designed to work with mainstream search engines such as Google, Bing, and Yahoo; so users search as normal, on their favourite search engines, but benefit from search recommendations from people they trust. The HeyStaks system has been described previously in [3], where the focus was on a description of its recommendation technique. The aim of this paper is to investigate the role of a novel reputation model during recommendation, whereby the search reputation of users is allowed to influence recommendation directly. We will return to the issue of reputation in following sections, but
first we present a brief review of HeyStaks in order to provide sufficient technical context for the remainder of this paper.

A. System Architecture

Figure 1 presents the HeyStaks architecture. There are two key components: a client-side browser toolbar/plugin and a back-end server. The toolbar serves a dual purpose: It provides users with direct access to the HeyStaks functionality, allowing them to create and share staks, tag or vote for pages etc. It also provides for the type of deep integration with mainstream search engines that HeyStaks requires. For example, the toolbar captures the routine search activities of the user (query submissions and result click-thrus) and it also makes it possible for HeyStaks to augment the mainstream search engine interface so that, for example, HeyStaks’ recommendations can be integrated directly into a search engine’s results page. The toolbar also manages the communication with the back-end HeyStaks server. Search activities (queries, click-thrus, tags, votes, shares etc.) are used by the server to update the HeyStaks stak indexes. And these stak indexes provide the primary source of recommendations so that when a user submits a query to a mainstream search engine, this query is fed to the HeyStaks server in order to generate a set of recommendations based on the target stak and, possibly, other staks that the user has joined.

B. The Recommendation Engine

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, \( S = \{r_1, ..., r_k\} \). Each result is also anonymously associated with a number of implicit and explicit interest indicators, based on the type of actions that users can perform on these pages, which include:

- **Selections (or Click-thrus)** – a user selects a search result (whether organic or recommended);
- **Voting** – a user positively votes on a given search result or the current web page;
- **Sharing** – 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** – the user chooses to tag and/or comment on a particular result or web page.

Each of these actions can be associated with a degree of confidence that the user finds the page to be relevant for a given query. Each result page \( r^S \) from stak \( S \), is associated with these indicators of relevance, including the total number of times a result has been selected \( (sl) \), the query terms \( (q_1, ..., q_n) \) that led to its selection, the terms contained in the snippet of the selected result \( (s_1, ..., s_k) \), the number of times a result has been tagged \( (tg) \), 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 indicated by Equation 1.

\[
r^S_i = \{q_1,...,q_n, s_1,...,s_k, t_1,...,t_m, v^+, v^-, sl, tg, sh\}. \tag{1}
\]

Importantly, this means 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 (http://lucene.apache.org) index, with each result indexed under its associated query and tag terms, and this 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, the searcher’s query \( q_T \) and current stak \( S_T \) are used to generate a list of recommendations to be returned to the searcher. For the purpose of this paper we will discuss recommendation generation from the current stak \( S_T \) only, although in practice 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 \( S_T \) by querying the relevant Lucene index using the target query \( q_T \). This effectively produces a list of recommendations based on the overlap between the query terms and the terms used to index each recommendation (query, snippet, and tag terms). Next, these recommendations are then filtered and ranked. Results that do not exceed certain activity thresholds are eliminated as candidates; such as, for example, results with only a single selection or results with more negative votes than positive votes (see [3]). Each remaining recommendation candidate \( r \) is then ranked according to a weighted sum of its relevance \( (rel) \) and reputation \( (rep) \) scores at time \( t \) as per Equation 2; where \( w \) is used to adjust the relative influence of relevance and reputation.

\[
\text{score}(r, q_T, t) = w \times \text{rep}(r, t) + (1 - w) \times \text{rel}(q_T, r) . \tag{2}
\]

The relevance of a result \( r \) with respect to a query \( q_T \) is computed based on Lucene’s standard TF-IDF metric as per Equation 3; TF-IDF is a well-known information retrieval term-weighting function [12] that 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.

\[
\text{rel}(q_T, r) = \sum_{\tau \in q_T} \text{if}(\tau \in r) \times \text{idf}(\tau \in r)^2. \tag{3}
\]
The above relevance model pays no attention to the source of the recommendation; i.e. the users who originally contributed the page to a stak or whose subsequent activities resulted in the page being recommended. For this reason recent research has looked at the possibility of adding a reputation component to recommendation. Recommendation candidates can be scored by a combination of relevance and reputation according to Equation 2, so that pages that have been contributed by many reputable users are considered more eligible for recommendation than those that have been contributed by fewer, less reputable users.

The key idea in the user reputation model for HeyStaks is that reputation can be calculated by mining the implicit collaborations that occur between users as a result of their searches. If HeyStaks recommends a result to a researcher, and the researcher 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 the user reputation model is that this act of implicit collaboration between producer and consumer confers some unit of reputation on the producer (Figure 2(a)), and to calculate the overall reputation of a user (producer) we need to aggregate these units of reputation across past collaborations.

The reputation model calculates the reputation of a producer as a weighted sum of the collaboration events in which they have participated. The simplest case is captured by Figure 2(a) where a single producer participates in a collaboration event with a given consumer, benefitting from a single unit of reputation as a result. More generally, at the time when the consumer acts (selects, tags, votes etc.) on the recommended result, there may have been a number of past producers who each contributed part of the search knowledge that caused this result to be recommended. 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 2(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 $S_j$.

$$\text{producers}(r, t) = \{p_1, ..., p_k\} \quad .$$

Then, for each producer of $r$, $p_i$, we update its reputation as in Equation 5. Reputation is shared equally among its $k$ contributing producers.

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

In this way reputation is based on the accumulation of collaboration instances and essentially the reputation of a user is a weighted sum of the number of collaborations they have contributed to by way of their past searches. Importantly this is just one specific way to model reputation according to this producer-consumer model; there are many other ways to count and accumulate reputation and in [13] we describe and evaluate a variety of different approaches.

In this paper, however, we assume the above user reputation model and focus instead on how it can be incorporated into the HeyStaks recommendation model. To do this we need a way to translate user reputation scores into corresponding page reputation scores which we consider in the following section.

### IV. Reputation as Collaboration

Fig. 2. 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 ($\text{rep}$) 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 $\text{rep}/k$ units of reputation.

### V. Page Reputation Models

In this section we describe a number of approaches to model page reputation. In each case, the goal is to calculate the reputation score of a result page $r$ at time $t$ based on the reputation scores of the page’s producers at that point in time; see Equation 6.

$$\text{rep}(r, t) = f(\text{rep}(p_1, t), ..., \text{rep}(p_k, t)) \quad .$$

For the purpose of illustration we will calculate each reputation score based on a hypothetical recommendation scenario for a page $r$ which is associated with a set of 10 producers with the following reputation scores at time $t$: $\{0.003, 0.014, 0.023, 0.052, 0.089, 0.097, 0.154, 0.297, 0.348, 0.581\}$. This includes a cross section of producers including some with low reputation scores and some with high scores. Note that in practice producer reputation scores are first normalised by the maximum producer reputation in the corresponding stak to ensure a score between 0 and 1.

#### A. Median Reputation

Perhaps the simplest way to translate user reputation into page reputation is to calculate the average reputation of the page’s producers. We propose to do this by finding the median reputation of the producers as follows:

$$\text{rep}(r, t) = \text{median}(\text{rep}(p_1, t), ..., \text{rep}(p_k, t)) \quad .$$


The advantage of this approach over a simple mean reputation is that the median statistic tends to better represent the central tendency of the set of user reputations. In the case of our hypothetical recommendation scenario the reputation of the page $r$ is 0.093 according to this median model.

**B. Max Reputation**

Another simple way of scoring a page based on the reputation of its producers is to take the maximum reputation value from that set. Formally, Max Reputation is calculated thus:

$$ rep(r, t) = \max (rep(p_1, t), ..., rep(p_k, t)) . \tag{8} $$

Scoring pages in this way is advantageous as the reputation of a page will not be harmed if, for example, many new, not yet reputable users have selected the page. In our recommendation scenario, the reputation of page $r$ is 0.581 by this approach.

**C. Harmonic Mean Reputation**

Harmonic Mean is an average measure that tends towards the lower bound of a set of numbers, and thus is more conservative than arithmetic mean or median. It is calculated by finding the reciprocal of the arithmetic mean of the reciprocals. Formally, the harmonic mean of a set of user reputation scores is calculated as:

$$ rep(r, t) = \frac{k}{\sum_{i=1}^{k} \frac{1}{rep(p_i, t)}} . \tag{9} $$

In this case, the reputation of page $r$ is 0.020. Harmonic mean may be a good indicator of the utility of a page in the sense that a page is only as reputable as its least reputable producer. However, rather than simply using the minimum producer reputation score available, harmonic mean permits the full range of producer reputation scores to influence the overall page reputation.

**D. Root Mean Square Reputation**

In order to get a sense of the magnitude of a set of user reputation scores, it is useful to calculate the root-mean-square (RMS) of those values. RMS is defined as the square root of the mean of the squares.

$$ rep(r, t) = \sqrt{\frac{\sum_{i=1}^{k} \left(\frac{1}{rep(p_i, t)}\right)^2}{k}} . \tag{10} $$

As RMS is a measure of the magnitude of a set of numbers, this value tends towards the upper bound of a set of user reputation scores. As such, RMS can be considered as a less conservative method of calculating page reputation compared to the previous harmonic mean approach. As per our worked example, the reputation of page $r$ using RMS is 0.243.

**E. Hooper’s Reputation**

In order to reinforce a page’s reputation according to the number of producers, keeping in mind their score, a different approach is required. A simple technique is George Hooper’s Rule for Concurrent Testimony, originally proposed as a technique to calculate the credibility of human testimony [14]. This is applicable in our case in the sense that users who have produced a result in HeyStaks are endorsing it, in the same way that a group of witnesses might attest to the same report. Hooper gives to the report a credibility of $1 − (1 − c)^k$, assuming $k$ reporters, each with a credibility of $c$ (where $0 \leq c \leq 1$). For HeyStaks, the quality of a page can be determined by performing the same calculation across the reputation scores if its producers.

$$ rep(r, t) = 1 − (1 − c)^k . \tag{11} $$

As per [14], we can calculate the reputation of result $r$ as $0.003 + (1 − 0.003)0.014 + (1 − 0.003)(1 − 0.014)0.023 \ldots$ and so on. The reputation of this particular result $r$ is 0.865.

In the following section we evaluate these five models by examining their influence on recommendations made by HeyStaks, according to Equation 2. Of course, the success of each model is determined by the extent to which they improve the effectiveness of the HeyStaks recommendation engine, which we analyse using a set of queries submitted to HeyStaks during the course of a live-user trial.

**VI. Evaluation**

In previous work [3] we have demonstrated how the standard relevance-based recommendations generated by HeyStaks can be more relevant than the top ranking results delivered by Google. In this work we wish to compare this relevance-based recommendation technique to an extended version of HeyStaks that also includes page reputation.

The purpose of this paper has been to propose a number of alternatives to calculating the reputation of content based on that of its producers who are helping other users (consumers) to search within the HeyStaks social search service. The hypothesis is that by allowing reputation, as well as relevance, to influence the ranking of result recommendations, we can improve the overall quality of search results. In this section we evaluate our page reputation models using data generated during a recent closed, live-user trial of HeyStaks, designed to evaluate the utility of the HeyStaks brand of collaborative search in fact-finding information discovery tasks.

**A. 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 [15] for a list of questions used in the trial. Each user was allocated a desktop computer with the 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. 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. Although trial users were made fully aware of how to use HeyStaks and its benefits, they were not explicitly encouraged to use the system during the trial, nor were they encouraged to perform one activity over another (sharing over tagging, for example).

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 [15]. For example, 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 how we can utilise reputation to measure the quality of recommendations. No reputation model was used during the live-user trial and so recommendations were ranked based on relevance only. However the data produced makes it possible for us to effectively replay the user trial so that we can construct our user reputation model and test each page reputation model by using each to re-rank HeyStaks recommendations. In order to ensure accuracy, we calculate the reputation of each recommendation made during the trial according to the reputation of its producers at the time the recommendation was made. 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 an explicit or implicit reference to an answer) by experts.

B. User Reputation

To get a sense of how users were scored by our model, we first examine the type of user reputation values that are generated from the trial data. It should be noted that each trial user was a member of only one stak, and thus could only receive (as a consumer) recommendations resulting from the activities of fellow stak members, and likewise could only gain reputation (as a producer) from the consumer activity of other members of the stak. Figure 3 shows box-plots displaying user reputation scores for each of the 4 shared staks at the end of the trial. We see 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. Despite the most reputable user in the trial hailing from the large 19-person stak, it is the 9-person stak which has the highest median reputation score. The interquartile range of reputation scores in this stak was relatively small, indicating that these users worked closely in collaboration with each other during the trial. It was these users who collectively outperformed most other users in the quiz, scoring higher marks and, on the whole, achieving a better correct answer per query rate (see [15] for more details). The box-plots show that there is a wide variation in reputation scores: Some users, particularly in the stak with 5 and 25 members, achieved an almost negligible amount of reputation. On the other hand, others received a score in excess of 20, the most reputable user scoring 37. These users were the primary drivers of search collaboration during the quiz.

C. Evaluating Page Reputation Models

The true test of the reputation models in this work 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. For the purpose of this evaluation we regenerate each of the recommendation lists produced during the trial using each of the page reputation models, based on the user reputation scores calculated at the appropriate point in time. 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 page reputation model we compute an overall relevance rate, as the ratio of the percentage of recommendation sessions where the top result was deemed to be relevant, to the percentage of those where the top result was not-relevant. Moreover, we can compare this to the relevance rate of the recommendations made by the standard HeyStaks ranking (i.e. when $w = 0$ in Equation 2) during the trial to compute an overall relevance benefit;

Fig. 3. User reputation score per user, per stak.
such that a relevance benefit of 40%, for a given reputation model, means that this model generated 40% more relevant recommendations than the standard HeyStaks ranking scheme.

Figure 4 presents a graph of relevance benefit versus the weighting \((w)\) used in Equation 2 to adjust the influence of term-based relevance versus page reputation during recommendation. Remember that as a recommendation is made, its reputation score is calculated based on the reputation of its producers at the time the recommendation is made. A similar trend can be seen in the figure for each technique in that relevance benefit is observed to increase initially with \(w\) before decreasing as \(w\) approaches 1. Peak performance occurs at different weights for each technique. For example, Harmonic Mean peaks at \(w = 0.2\) before dipping below 0% relevance benefit for all remaining weights. Hooper, the best performing technique by some distance, peaks twice at \(w = 0.4\) and \(w = 0.8\), each time achieving around 55% relevance benefit. Four of the five techniques result in considerable improvement over the standard HeyStaks relevance-based model, each achieving at least 30% relevance benefit at their respective optimal weighting. Harmonic Mean was the worst performer, only managing a maximum of 3.4% relevance benefit, and for most weights it delivered a negative relevance benefit, meaning the standard HeyStaks recommendation engine delivered proportionately more relevant results across the trial. This is most likely due to the fact that harmonic mean tends towards the lower bound of a set of producer reputation scores; if, for example, a page that has been produced by many users with varying reputation, the harmonic mean of those scores will tend towards the lower bound of the set. Conceptually, we may not wish to punish pages whose producer reputation scores have high variance, particularly where some producers have high reputation scores. Hooper achieves the best relevance benefit of all, and at 55% for \(w = 0.4\), represents a realistic option for integration into a live HeyStaks reputation engine. This may be the most suitable option as the score it produces for a page is a consensus based on the reputation of its producers. The technique promotes the idea that a page will have a high score by way of reinforcement from its producers, assuming they are reputable.

Figure 5 shows the median relevance benefit across weights for each page reputation model. A Kruskal-Wallis test indicates that there are statistically significant differences between the performance of the reputation models at the .01 level. Examining the pairwise differences between models, Tukey’s Range test indicates that there are significant differences between Hooper’s technique and two other techniques at the .05 level, Harmonic Mean and Root Mean Square. These findings further highlight the strong performance achieved by the Hooper page reputation model.

D. Limitations

In the above we have compared a number of page reputation models to map user reputation scores onto the content users produce, based on real-user search data. One limitation of this approach is that although the evaluation uses trial-user data, the final recommendations are not themselves evaluated using the trial users. Instead we replay users’ searches to generate reputation-enhanced recommendations. The main reason for this is the difficulty in securing sufficiently many 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 recommendations 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 genuine 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 ranking signal. Of course this evaluation cannot tell whether users would actually select these reputation-ranked recommendations, although there is no reason to believe that they would treat these recommendations differently from the default HeyStaks recommendations, which they are inclined to select. We view this as a matter for future work.

Another point worth noting is that the user trial is limited to a specific type of search task, in this case a Q&A search task. As such 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 [3] has examined the role of HeyStaks in more open-ended search tasks, where its default relevance-based recommendations were also found to be beneficial to end-
users. As part of our future work we are currently in the process of deploying and evaluating our reputation models across similar general-purpose search tasks.

VII. Conclusions

This paper is about social search and broadly speaking the research looks at how we can make web search more collaborative. In particular we describe the HeyStaks social search platform which brings collaborative web search to mainstream search engines such as Google, Bing, and Yahoo, via browser plugins. The specific contribution of this paper is the introduction of page reputation models that can be used to influence result recommendations made by HeyStaks at search time, recommendations that originate from the past searches of communities of collaborating searches. The HeyStaks user reputation model can be used to model the effectiveness of a user from a search standpoint. For example, users whose search results are frequently recommended to, and acted on by, other users are considered to be reputable searchers. The intuition behind this work is that these reputation scores can be viewed as a type of evidence in support of page recommendations so that recommendation candidates that come from many reputable users (producers) are considered to be more reliable that recommendations from less reputable users.

In this paper we have explored different ways to translate user reputation scores into an overall page reputation/relevance score. We have described the results of a comparative evaluation in the context of real-user data which highlights the ability of these techniques to improve overall recommendation quality, when combined with the relevance-based recommendation ranking metrics that are currently used by HeyStaks. For example, many of the page reputation models can improve recommendation relevance (compared to the standard HeyStaks benchmark) by over 30%. Moreover, we have found that one model, based on Hooper’s rule for Concurrent Testimony [14] is capable of delivering relative improvements of up to 55%. 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.

In this paper we have focused on the role of reputation during the recommendation process, in order to maximise the relevance of the community recommendations made by HeyStaks. But this is just one use of reputation in a system such as HeyStaks. For example, in many social systems there is the risk that malicious users will attempt to manipulate the outcome of social processes; see relevant work in recommender systems research [16]–[18]. In HeyStaks, for example, it is possible to malicious users to flood search staks with irrelevant or self-interested results, which could impact recommendation quality. By using reputation to mediate recommendation it will be possible to guard against this; these malicious users will have low reputation scores (assuming their contributions are rarely acted on by other users) and as such their contributions will be unlikely to appear in future recommendation sessions. Furthermore, reputation can be exposed to users of systems like HeyStaks as an important social signal. For example, although HeyStaks’ recommendations are anonymous (so users do not know the source of result recommendations at search time) it may make sense to explain recommendations with reference to the reputation of producers in the future.

ACKNOWLEDGMENT

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

REFERENCES

PAPER 5: A COMPARATIVE STUDY OF REPUTATION MODELS FOR SOCIAL RECOMMENDER SYSTEMS

Kevin McNally, Michael P. O’Mahony and Barry Smyth
Published in User Modeling and User-adapted Interaction (2013), pp 1–42
A Comparative Study of Collaboration-based Reputation Models for Social Recommender Systems

Kevin McNally · Michael P. O’Mahony · Barry Smyth

Received: date / Accepted: date

Abstract Today, people increasingly leverage their online social networks to discover meaningful and relevant information, products and services. Thus, the ability to identify reputable online contacts with whom to interact has become ever more important. In this work we describe a generic approach to modeling user and item reputation in social recommender systems. In particular, we show how the various interactions between producers and consumers of content can be used to create so-called collaboration graphs, from which the reputation of users and items can be derived. We analyze the performance of our reputation models in the context of the HeyStaks social search platform, which is designed to complement mainstream search engines by recommending relevant pages to users based on the past experiences of search communities. By incorporating reputation into the existing HeyStaks recommendation framework, we demonstrate that the relevance of HeyStaks recommendations can be significantly improved based on data recorded during a live-user trial of the system.

Keywords Reputation · Social Recommender Systems · Collaboration Graphs

1 Introduction

As the online world has matured, the nature of the world-wide web has evolved. During the 1990s the web was about pages and links. Today it is as much about communities and

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

Kevin McNally
CLARITY Centre for Sensor Web Technologies
School of Computer Science and Informatics, University College Dublin, Ireland
E-mail: kevin.mcnally@ucd.ie

Michael P. O’Mahony
CLARITY Centre for Sensor Web Technologies
School of Computer Science and Informatics, University College Dublin, Ireland
E-mail: michael.omahony@ucd.ie

Barry Smyth
CLARITY Centre for Sensor Web Technologies
School of Computer Science and Informatics, University College Dublin, Ireland
E-mail: barry.smyth@ucd.ie
Indeed, over the past decade the power of the web as a collaboration platform has been dramatically demonstrated time and time again, and its collective wisdom has been applied to a diverse set of challenging scenarios, from the indexing of images to the creation of shared repositories of authoritative content (Lih, 2004; von Ahn, 2006; Siorpaes and Hepp, 2008; Hacker and von Ahn, 2009; Law and von Ahn, 2009; Walsh and Golbeck, 2010; De Alfaro et al, 2011; Goh et al, 2011; Guy et al, 2011). Importantly, the web, which heretofore created a network of content, has now matured to accommodate a network of relationships. Today most online users are connected to a variety of other people, near and far, by a complex and mature social graph, whether to our friends and classmates via Facebook, colleagues via LinkedIn, or our peers through services like Twitter.

In the early days of the web of content, the ability to evaluate content quality rapidly became a pressing need in order to help people locate the right information at the right time. Perhaps the single most important innovation of the web of content was the development of techniques for rating the authority of content in order to filter and rank pages during web search (Brin and Page, 1998; Kleinberg, 1999). This led directly to the age of web search and the success of search engines like Google. As the social web matures, and as our social graphs explode, we are faced with a similar need: the need to filter and rank users and relationships, which forms the basis of our social graph, which is increasingly playing the role of the key information filter in the social web of shared knowledge. As a result, researchers have begun to explore the idea of trust and reputation on the social web, as an analogue for authority on the web of content (Kempe et al, 2003; Golbeck and Hendler, 2004; Zeng et al, 2006; Jörsang et al, 2007; Kuter and Golbeck, 2007; Wu and Tsang, 2008; O’Donovan, 2009; Duan et al, 2010; Lazzari, 2010; McNally et al, 2011; Kuter and Golbeck, 2010; Weng et al, 2010; Bakshy et al, 2011; Canini et al, 2011; Recuero et al, 2011; Pal and Counts, 2011; Cai et al, 2011). In this work we are also interested in reputation in the social web, and the main contribution of this work is a description and evaluation of a set of reputation models that can be used to estimate the reputation of collaborating users and to use this information to infer the quality of the content or items that they interact with.

The perspective that drives this work is that the social web, at its heart, is about collaboration in the broadest sense of the word. For example, when a group of users edit a Wikipedia page they engage in a form of collaboration. Collaboration also occurs when an Amazon shopper acts on a recommendation derived from the ratings of other users. And when a user re-tweets another user we again see a form of collaboration in action. These are all examples of a collaboration event. They can occur anonymously as with Amazon and often Wikipedia, or we might know our collaborators, as with Twitter. Consequently, the key idea behind our reputation framework is that that the reputation of a user is based on the collaboration events that they participate in, and these collaboration events are the fundamental unit of reputation.

Very briefly, each collaboration event involves at least two users, a so-called producer and a consumer and the event represents some action by the consumer on an item/piece of content/asset contributed by the producer. For example, when Bob edits a Wikipedia article created by Mary, Bob is the consumer, Mary is the producer and Bob is effectively collaborating with Mary, albeit implicitly and without the need for explicit coordination beyond the collaboration platform provided by Wikipedia. Likewise, when Tom re-tweets a message posted by Suzie, a form of collaboration is occurring, with Tom the consumer and Suzie the producer. Of course, consumers may in turn become producers; if Tom’s re-tweet is again re-tweeted by Alan then Tom is now a producer and Alan a consumer.

As we shall see this collaboration perspective means that we can represent the activities of users as a type of collaboration graph, the nodes of which represent users, and with
directed edges connecting producers to consumers. As new collaboration events occur, additional nodes and edges are created. Reputation can be inferred by examining the structure and dynamics of this collaboration graph. As collaboration events occur weighted edges from consumers to producers are inserted, with reputation accumulating at the graph nodes. In previous work we have presented a proof-of-concept of this approach to reputation modeling, focusing on one particular approach to accumulating reputation (McNally et al, 2010, 2011). In this paper we broaden this perspective in a number of important respects. Firstly we generalise our approach and describe a number of user reputation models based on the collaboration graph. In turn, we describe how these models can influence the recommendation of information to users as part of a recommender system, and evaluate a variety of different item reputation models that form the basis of this approach to recommendation.

While in this paper we are primarily concerned with computational models of reputation applied in an online setting, there is a long history of analogous work, particularly in the sociology literature, when considering the nature and role of trust and reputation in communities and societies; (Raub and Weesie, 1990; Mayer et al, 1995; McKnight and Chervany, 1996; Rousseau et al, 1998; Gambetta et al, 2000). We draw on this work, as well as related work in the online space, to evaluate this reputation framework in the context of a collaborative approach to Web Search (Spertet and Gauch, 2005; Morris and Horvitz, 2007a; Smyth, 2007; Morris and Horvitz, 2007b; Amershi and Morris, 2008). Our target system, HeyStaks, provides a novel approach to web search by adding a collaboration layer to mainstream search engines via a browser plugin. Briefly, HeyStaks facilitates implicit collaboration between searchers (Smyth et al, 2009). For example, when Nicola searches, in addition to Google’s mainstream search results, she may receive recommendations from her social graph; these recommendations are results that other searchers have found to be relevant for similar queries. If Nicola selects one of these recommendations she is effectively engaged in a collaboration with the original searcher. In this paper we describe how our model for calculating reputation can be incorporated into HeyStaks’ default recommendation engine and show that it has the potential to significantly enhance the quality of recommendations by evaluating it on live-user search data.

The remainder of this paper is organised as follows. In Section 2 we review recent related work on trust and reputation in the social web, highlighting a number of different approaches to measuring, analysing, and utilising this type of information across a variety of online services. Following this, in Section 3, we describe our reputation framework in detail, formalising the notion of collaboration, producers and consumers, and describing how the collaboration graph can be constructed based on the interactions between users in online social platforms. Section 4 describes a number of user reputation models that are derived from the collaboration graph, and how user reputation can be used to create models of item reputation. We go on to explain how these item reputation models can be applied to existing recommendation systems, effectively providing a complementary dimension to user/item similarity during recommendation. In Section 5 we describe the HeyStaks social search system as a test-bed for our approach to modeling user and item reputation. Finally, we give a detailed account of a live-user trial in Section 6, the data from which serves as the raw material for our reputation analysis and evaluation. We demonstrate, for example, that reputation can be captured effectively by the models described herein, and we show how different approaches to harnessing this reputation can positively influence recommendation quality. Finally, having explored this reputation design space, we go on to highlight how a specific reputation model instance stands out in terms of model performance and recommendation efficacy.
2 Background

In the early days of the web, the proliferation of online content meant it became increasingly difficult for people to distinguish between what content was useful and what was useless or even harmful. The work of Brin and Page (1998) and Kleinberg (1999) was significant in understanding that link connectivity can provide a very useful signal when it comes to the relative importance of web pages. Nowadays mainstream search engine technologies are largely based on this idea of connectivity. Around the same time, the popularity of online marketplaces exploded: Online platforms that allowed business-to-consumer or consumer-to-consumer transactions to take place became a key service provided by the web. Whereas traditionally interaction in business took place face-to-face, sites such as eBay and Amazon began to allow people to interact with each other remotely and even without interacting users knowing each other in the offline world. This presented a problem: How do buyers know which sellers are reliable and which are not? Where Google was effectively differentiating between reliable and unreliable pages, these online marketplaces needed a mechanism that distinguished reliable from unreliable users (Dellarocas, 2003).

2.1 Trust and Reputation in Societies

As mentioned previously, reputation, influence and trust have long been a topic of research within the sociology community as researchers have considered the origins of reputation, and the nature of trust between individual groups within societies. For example, early work by Raub and Weesie (1990) examined the origin of reputation within communities, highlighting the emergence of reputation when community members are knowledgeable of past actions. Moreover this work studied the effect of reputation on the efficiency of community interactions, concluding that knowledge of past performances drive reputation and facilitate more efficient interactions. Conversely they found that delays in the dissemination of this knowledge negatively impacts interaction efficiency. Considerable attention has been devoted to understanding the related matter of the trust between users in social contexts. For example, Gambetta et al (2000) explored the nature of trust in the context of social cooperation. Gambetta argues that when we say someone is trustworthy, we implicitly mean that the probability that he will perform an action that is beneficial, or at least not harmful to us, is high enough for us to consider engaging in some form of cooperation with him. As a result our beliefs about an individual can have greater importance than our motives for cooperation.

Mayer et al (1995) further unpacks the nature of trust relationships between individuals in cooperative contexts, by examining related factors such as confidence, predictability, and risk. Moreover, this work highlights the important issue of context in a trust-based relationship and sensitivity of trust to changes in context. An individual might trust another in a certain context but not in a different context, for example. Similar ideas are explored by Rousseau et al (1998), who also propose that there is no single definition of trust, that it is a concept that is always in transition, and that it evolves as our interactions with others evolve; see also the work of McKnight and Chervany (1996).
2.2 Defining Online Trust and Reputation

The main focus of this work, however, is on reputation in the online world. Much early research articulated the difficulty people have in determining the reliability of others in online scenarios (Jones et al, 2000; Olson and Olson, 2000; Resnick et al, 2000; Shneiderman, 2000). It focused on the idea that the system itself could not make the necessary decisions. Reliability of users was subjective and often relied on external factors (such as whether both participants in an online transaction behaved honorably). The online community itself should be allowed to determine who would or would not cause harm to others in the future. The research instead placed emphasis on the importance of facilitating the development of trust-based relationships between users. One user trusts another if they believe that any future transaction will be rewarding rather than detrimental (Golbeck and Hendler, 2004). However, how can two people make this determination if they have never interacted with each other before? There is a need for trust information to be publicly accessible so users can draw on the experiences of others. In order for a user to trust a stranger, they need to know how reputable they are (Resnick et al, 2000).

A key idea in this work is that the reputation of a single user can be deduced from looking at the outcome of transactions they have taken part in. If a user has made good on transactions with others in the past, it is likely they will continue to do so, and thus they are reputable in their community. The discussion on reputation put forth by Jøsang et al (2007) speaks to this definition: “Reputation can be considered as a collective measure of trustworthiness (in the sense of reliability) based on the referrals or ratings from members in a community.” O’Donovan and Smyth (2005) come to a similar conclusion about how reputation can be couched in terms of trust. They believe that reputation is a function of trust, and both can be computed over time. From looking at this early work a view of the relationship between the concepts of online trust and reputation becomes clear: Trust occurs between a pair of users and is brought about by one or both users performing well in the context of some social or transactional contract. For example buyers and sellers will come to trust each other on eBay if a given transaction goes to plan. But if one or more parties break this implicit contract then trust will break down. It improves as more positive interactions take place between the pair. The reputation of a user can be seen as an aggregation of the trust that they have established with the set of users they have interacted with in the past. This research strongly indicates that although online trust and reputation can be distilled as concepts in their own right, they are closely interlinked, as explained by Mui et al (2002).

2.3 Transactional Feedback

Measuring and communicating this kind of reputation allows new users to base their decisions on the experiences of others. For example a new eBay buyer will trust a seller if there is information available that indicates this seller has performed well in a similar role in the past, even though the new buyer has never dealt with this seller themselves. However, in order to measure trust and reputation, they must be established as quantifiable entities. Early online reputation systems aggregated information users gave about seller performance and explicitly displayed it as an overall reputation score (Resnick et al, 2000). These early systems, largely employed by e-business websites, were based on aggregating buyer feedback data and displaying it as part of a seller’s profile. The reputation system employed by eBay was one of the earliest widely popular systems of its kind.
This feedback-based mechanism has been examined by Resnick and Zeckhauser (2002) and later by Jøsang et al (2007). The system 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. An example profile page showing seller reputation can be seen in Figure 1. However, 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 always a reliable indicator of future vendor performance. Jøsang et al (2007) confirms this, stating such systems require manual curation and protection from malicious users. For further discussion on how trust and reputation information can enhance system robustness, see Jøsang and Golbeck (2009); Lazzari (2010).

Since the development of these simple feedback-based mechanisms, work has been conducted to investigate how trust information can be utilized to improve specific platforms in different ways, for example by improving the quality of recommendations made by collaborative filtering systems (Massa and Bhattacharjee, 2004; O’Donovan and Smyth, 2005; Massa and Avesani, 2007; Kuter and Golbeck, 2010), enhancing the robustness of collaborative content websites like Wikipedia (Chatterjee et al, 2008; De Alfaro et al, 2011) and, more recently, incentivizing participation in online social networks (Yang et al, 2008; Lazzari, 2010; Li et al, 2011). In each case the interaction between users occurs in the context of some item or service or piece of content, and the quality of this interaction is measured as the basis of trust. Some of these systems focus only on calculating trust scores between pairs of users (O’Donovan and Smyth, 2005), some propagate these trust scores across a network (Guha et al, 2004; Kuter and Golbeck, 2010), and some aggregate these scores to achieve an overall reputation score for each individual (De Alfaro et al, 2011; Li et al, 2011). Each
method for gathering trust or reputation information is platform-specific, but all gather information by examining user-user interaction in the system being augmented. Specifically, one user provides some piece of information or service, and another user takes action on either the producing user or the produced item, from which trust information can be inferred.

2.4 Trust in Recommender Systems

In some cases it is not possible to gain trust information from the user directly, and many systems that rely on user interaction do not provide feedback functionality, so statements of trusts must be derived from implicit, indirect actions. An example of a mechanism for measuring trust based on implicit, indirect user interaction is given in the work of O’Donovan and Smyth (2005). The authors address the idea of interpersonal, context sensitive trust in recommender systems. Trust between users is not calculated by examining feedback received directly from users. Instead it is inferred that one user trusts another if their ratings can be used to reliably predict the other’s rating for an item. The 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. O’Donovan and Smyth posit that trust can be estimated by measuring the accuracy of a profile at making predictions over time. Using this approach average prediction error is improved by 22% on standardized test sets.

Using socially-generated information such as the content of reviews and genre information to complement collaborative filtering systems was introduced by Basu et al (1998). This idea has been extended to infer trust and reputation information by augmenting the interface to allow users to directly and explicitly give feedback to other users, then aggregating this feedback in some way. Golbeck (2006) developed a trust-based recommender in which neighbours are not selected based on ratings similarity but rather based on explicitly provided trust data. Similar to Golbeck (2006), Avesani et al (2005) propose a trust algorithm called MoleTrust that can be used to augment an existing collaborative filtering system. The mechanism calculates a “trust metric” based on explicit, direct feedback given by users, which propagates across a network of content producers. This algorithm can be tuned to propagate over a specific depth across a social graph, controlling the prediction that one user trusts another even though they may have not explicitly interacted. They find that MoleTrust can improve the accuracy of predictions made by a recommender system, even in cases where users have provided few ratings. The authors provided similar work using data gained from the Epinions\textsuperscript{1} consumer review website, a platform that again allows users to give explicit trust information regarding other users in the system (see Massa and Bhattacharjee, 2004; Massa and Avesani, 2007). The ideas put forward by the creators of MoleTrust have since been extended by Kuter and Golbeck (2010) to introduce a locally determined trust model known as SUNNY which uses a Bayesian approach to assign trust scores to a network of users. The algorithm utilizes explicit trust information provided by users of the FilmTrust network\textsuperscript{2}.

\textsuperscript{1} http://www.epinions.com.
\textsuperscript{2} http://trust.mindswap.org/FilmTrust.
2.5 Reputation and Collaborative Content Creation

Although collaborative filtering systems involve collaboration between users, the content with which the users interact is often not created by the users themselves (e.g. books, movies, music etc.). In scenarios where users do create content for public consumption, often finding the consensus is vitally important to ensure items of content are of the highest possible quality. The online encyclopedia Wikipedia\(^3\) allows for this kind of collaborative creation of content. Due to the popularity of the site – the website’s steady growth has resulted in the creation of nearly 4 million articles as of 2012\(^4\) – manual maintenance of each article is becoming increasingly unfeasible. As such, research has been conducted that explores the idea of applying a reputation model for both contributors and articles. However, Wikipedia does not provide content authors with the facility to directly communicate their level of trust in others contributing to the system.

Work by Zeng et al (2006), Chatterjee et al (2008) and De Alfaro et al (2011) propose a novel way to infer overall reputation scores by examining the revision history of articles. It is suggested the extent to which one user edits another’s content is in inverse proportion to the level of trust they have in that user’s content, and thus the user himself. The earlier work of Zeng et al (2006) focuses on calculating article trustworthiness, but De Alfaro et al (2011) extend the idea to model contributor reputation. Each contributor’s score, and thus the reputation of the content they provide, depends on how long their revisions survive before editing: If an article is fundamentally changed and that revision is accepted for a long amount of time, the reputation of the original author suffers while that of the reviser is boosted. This temporal, asymmetric aspect of the model promotes robustness as a user cannot instantly obtain high reputation, rather they have to earn it over time.

2.6 Trust and Reputation on the Social Web

Online interaction between users is nowhere more free than on today’s most popular Social Web platforms. Trust between users, and indeed user reputation, can be inferred in many different ways using any number of signals. Work has been carried out to propagate scores across a network of users where trust information is given explicitly and directly by users on websites such as Epinions (Guha et al, 2004) and more recently the professional social network Naymz\(^5\) (Lazzari, 2010). The author cautions that calculating reputation on a global level allows users who have interacted with only a small number of others to potentially accrue a high degree of reputation, making the system vulnerable to malicious use. Similar conclusions are made by Jøsang et al (2007), who suggest that vulnerability lies in the site itself, allowing malicious users to game the reputation system for their own ends.

Work conducted by Pal and Counts (2011) focuses on identifying topics on Twitter and ranking people according to their authority on that topic. They measure a number of different features related to a user’s account such as number of original tweets, number of links shared and number of topically active followers. They examine “re-tweet impact” – a term described as the ratio of tweets a user has re-tweeted to the number of times that user’s content has been re-tweeted by others. The authors’ algorithm clusters users by topic, then ranks users according to a set of metrics that might be a good indication of authority in

\(^3\) http://www.wikipedia.org.
that topic; for example, the number of times a user’s topically relevant tweets have been re-tweeted by others. Similar work was carried out by Weng et al. (2010), where twitter users are assigned a score based partly on the idea of homophily, specifically how many people they follow who also follow them. Topic-sensitive graphs are constructed based on the content of users’ tweets, and users are ranked according to the relative influence of others in their topic-based subgraph. Similar work has focused on homophily in large-scale instant messaging networks (Aral et al., 2009; Schaal et al., 2010). The latter work focuses on processing the implicit feedback between blogs and distinguishing between two types of signals per user with respect to their roles as authors and sources of feedback. More recently, Canini et al. (2011) have examined ranking users of Twitter based on users’ credibility per topic. This approach would allow users to input a search query, returning users who frequently tweet using those query terms, ranked by their credibility to the related topic. For similar work on influence on the Social Web, see Duan et al. (2010); Bakshy et al. (2011); Cai et al. (2011).

2.7 Summary Discussion

In summary, the motivation behind applying reputation systems, as evidenced by this recent work, is to incentivize users to engage online with others and thus with the platforms that employ such systems. This can be aided by users trusting the system (Joinson, 2008), but also by rewarding users with positive feedback (Cheng and Vassileva, 2005) or with improved social standing (Recuero et al., 2011). It stands to reason that, if successful, a reputation system can not only distinguish trustworthy users from untrustworthy ones, but also determine the quality of the resources users provide. At their core, many of these systems calculate trust or reputation by building a network based on some explicitly or implicitly gained information in a specific context. Each incident edge in the network tells us something about how one user trusts another. To date all of these related works have developed specialised trust/reputation models in a specific setting or context. A key idea that underpins many of these approaches is the idea of collaboration, either implicit or explicit. Even in occasions where trust scores are propagated throughout a network, some form of user collaboration is required to infer trust. In this paper, we propose a generalized model of reputation by building a graph based not on trust, but on user collaboration: If one user produces a piece of content that is subsequently consumed by another user, their reputation increases. In the following section we describe how the interactions of users in online social platforms can be captured using a collaboration graph and then discuss how reputation can be calculated by examining this graph. By using the social search utility HeyStaks as a case study, we show how the model can be leveraged to positively influence the relevance of recommendations made by HeyStaks to users.

3 Collaboration in the Social Web

The idea of online collaboration has led to the democratization of content creation and consumption. Whereas once people spent large sums of money to buy a large encyclopedia written by a few choice experts, now authors collaborate online to create articles on Wikipedia. Similarly, users are regularly sharing content of different types with each other; from images
on sites like Imgur\(^6\) and Flickr\(^7\), news articles on Digg\(^8\) and Reddit\(^9\), or their knowledge of specific concepts on the StackExchange Network\(^10\). Such behaviour is becoming increasingly commonplace due to the proliferation of social media platforms. For example, the introduction of Facebook’s “like” functionality has resulted in users endorsing content from their community 2.7 billion times daily, as of 2012\(^11\). Users are also producing content at staggering rates: over 200 million tweets are broadcast to the Twitter community on a daily basis\(^12\), and a recent study by Phelan et al (2011) has shown that around 22% of these contain a URL to offsite content. In a single month in 2010, the social news website Reddit saw users post over 350,000 pieces of content. These posts are ranked according to the number of positive and negative votes they receive from users – an explicit indication of their quality as perceived by the community. In that month users voted on content over 3.5 million times\(^13\). Users can collaborate in groups on a single piece of information that is then offered for public consumption. On Wikipedia\(^14\) users have created over 3.7 million articles in English alone, many of which have been revised and edited by multiple authors.

In the previous section we discussed various trust and reputation models in terms of how each system provides users with the means to communicate trust information. Pairs of users interact, mediated by some piece of information or content, and the consumer of that content gives feedback in some way, on the user or piece of information, and either explicitly or implicitly. The level at which these users interact determines the trust one user has in another, and the level to which a person performs in their community determines their reputation. We introduce a model that gathers reputation information which is applicable to any collaborative online environment, regardless of the nature of the feedback. Other systems may measure reputation by aggregating the level of trust between users (for example, Avesani et al (2005); Golbeck (2006)). Our model does not require explicit trust statements to be made between users to calculate overall user reputation. Instead, reputation is determined by measuring the level of collaboration that occurs between users.

In Section 4 we describe how we can use this collaboration model as the basis for calculating user and item reputation. In fact we will describe a number of different techniques for calculating reputation at the user and item level.

### 3.1 A Model of Online Collaboration

In what follows we will distinguish between trust (as a function of some user-to-user interaction) and reputation (as a function of a particular user) as discussed previously. In order to frame this research it is useful to consider two different dimensions by which we can understand the source and type of trust information used in many online social systems. Trust signals can be either implicit or explicit and they can be direct or indirect (see Figure 2) and different systems and approaches can be usefully compared along these dimensions. Examples of research previously discussed can be viewed in these terms; see Figure 3. In the case

---

\(^7\) http://www.flickr.com.
\(^8\) http://www.digg.com.
\(^12\) http://www.techcrunch.com/2011/06/30/twitter-3200-million-tweets/.
\(^13\) http://www.reddit.com/r/Redditresearch/comments/de4re/basic_frequency_plots_for_a_months_worth_of.
Fig. 2 Two different types of feedback from which trust can be inferred. Figure (a) shows a simple, direct model where user \( u_2 \) gives feedback directly to user \( u_1 \). Figure (b) depicts the inference of trust based on the feedback that user \( u_2 \) gives about item \( i \), rather than directly to user \( u_1 \) who produced the item.

of the former reputation information can be based on a direct user to user interaction, such as one user rating another user. Conversely reputation information may be derived indirectly if, for example, users interact by virtue of some item as is the case when users collaborate by editing a Wikipedia article.

Reputation information can also be based on implicit or explicit statements of trust. In eBay users provide explicit feedback on sellers or buyers; an example of a direct, explicit reputation model. In other scenarios reputation information is derived from implicit factors. In a recommendation scenario a user who receives and acts on a recommendation is implicitly acknowledging those similar users who served as the source of said recommendation; an example of an indirect, implicit reputation scenario because the users interact with respect to a separate recommended item. Likewise, when a user follows another user in Twitter we can view the user as displaying some level of trust on the followed user; this is an example of a direct, implicit reputation scenario.

Thus our approach to reputation acknowledges different sources and types of reputation information. In each case, however, the common denominator is an instance of collaboration between two users, which we view as a collaboration event. Such an event can occur directly or indirectly, and indeed explicitly or implicitly, as determined in the previous section. In fact, such collaboration can happen asynchronously, sometimes with users having interlinked but distinct goals. For example a user of the Stack Exchange network may answer another user’s question some time after being first posed. This answerer has a different goal to the questioner, but these two people have indeed collaborated in their sharing of information. And the final piece of our approach proposes two possible ways in which users can participate in these collaboration events (see Figures 4 and 5). Specifically in each collaboration event there is a \( \text{producer} \) and a \( \text{consumer} \). The producer \( p \) is the originator of some piece of information that is acted on, interacted with or consumed by the consumer \( c \).

More generally a collaboration event can refer to a single consumer and multiple producers. For example, in a recommender system a consumer’s selected recommendation may have come from many similar users. Similarly a user may read a tweet that has appeared on their Twitter feed as the result of a string of re-tweets by different users. Although these are cases where multiple users have produced a piece of information, this should still be viewed as a single instance of collaboration. Finally, the key idea in our approach is that these collaboration events can be scored in effect transferring a unit of trust from consumer to producer;
see Figure 4. Or, if there are multiple producers, then the unit of trust can be shared between the $k$ producers as in Figure 5. During a collaboration event, a single unit of trust is shared equally between producers, but in some circumstances trust could be weighted in favour of particular producers that have contributed more to this single collaboration. However, this is outside the scope of this paper, and we view that as an interesting matter for future work.

### 3.2 From Collaboration Event to Collaboration Graph

Over time sequences of collaboration events come to form a collaboration 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 trust scores flow along these edges. The trust score associated with each edge – based on the type of interaction and
domain – form the basis of user reputation which is accumulated at the nodes as per Section 4. For example, Figure 6 depicts a scenario where eight users of a system have interacted collaboratively with each other. Here it can be seen, for example, that user \( u_c \) has consumed information produced by users \( u_b \) and \( u_d \) and thus edges are drawn from \( u_c \) to each of these users with weights of 0.5.

So far we have described how we distill collaboration events and how each event can be interpreted as a set of directed, weighted edges in a collaboration graph. We can look at all incident edges per-user to get a sense of how users collaborate with not just another single user, but with their community. In that sense, the trust score illustrated in Figure 4 can be
viewed as a fundamental unit of reputation. In the next section we describe our approach for calculating user reputation based on users’ propensity to collaborate with others, as well as outlining a number of alternatives using well-known link analysis techniques.

4 Modelling Reputation

Given a collaboration graph, how can we calculate the reputation of the collaborating users? Moreover, how can we use this reputation to positively influence the future dissemination of content? Both of these questions are addressed in this section where we describe three user-based reputation models, as well as four approaches to calculating item reputation based on the scores of the producing users.

4.1 User Reputation

We now present a series of methods to calculate user reputation by examining the collaboration graph: The “Weighted Sum” model (McNally et al., 2010) and the two well-known link analysis techniques, PageRank (Brin and Page, 1998) and HITS (Kleinberg, 1999).

4.1.1 Reputation as Weighted Sum

The Weighted Sum model simply calculates the reputation of a producer $p_i$ at time $t$, $rep(p_i,t)$, as the sum of the weights of incoming edges as follows:

$$rep(p_i,t) = \sum_{e \in E_{p_i}} w_e$$  

where $E_{p_i}$ is the incoming edge set of producer $p_i$ and $w_e$ is the weight of edge $e$. A user may gain a high reputation score from assuming the role of producer in many collaboration events. However, the growth of this user’s reputation may be stymied if they are only one of many producers of a number of consumed items. This feature of the model may be used to encourage users to produce their own high-quality content, and thus discourage piggybacking on the work of other producers.

4.1.2 Reputation as PageRank

PageRank is the well-known algorithm used by Google to rank web search results (Brin and Page, 1998). 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 numbers 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 PageRank algorithm can be readily applied to compute the reputation of collaborating users, which take the place of web pages in the graph. When a collaboration event occurs, directed links are inserted from the consumer to each of the producers as described...
above. 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:

$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{|L(p_j)|},$$

(2)

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 $L(p_i)$ is 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 Python library NetworkX\textsuperscript{15} implementation of PageRank, which implements the algorithm according to Page et al (1999), and Langville and Meyer (2005).

### 4.1.3 Reputation as HITS

The HITS algorithm (Kleinberg, 1999) 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.

As with PageRank, we use the collaboration graph and the HITS algorithm to estimate user reputation. In this regard, it may at first seem appropriate to consider producers as authorities and consumers as hubs. However, as we will discuss in Section 6, 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 information. Thus we model user reputation using both authority and hub scores, which we compute using the NetworkX implementation of the HITS algorithm. This implementation is based on work by Kleinberg (1999), and Langville and Meyer (2005). Briefly, the algorithm operates as follows. After initialization, repeated iterations are used to update the authority ($auth(p_i)$) and hub scores ($hub(p_i)$) for each user $p_i$. At each iteration, authority and hub scores are given by:

$$auth(p_i) = \sum_{p_j \in M(p_i)} hub(p_j)$$

(3)

$$hub(p_i) = \sum_{p_j \in L(p_i)} auth(p_j)$$

(4)

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

In the above we have described reputation models for users. People accumulate reputation when content that they have produced is consumed in some way by other users. We have described how reputation is distributed between multiple producers during these collaboration events. In the following section we describe how this reputation information can be used to capture the quality of content these users produce. In order to achieve this, we propose a number of methods to map user to item reputation by considering the source of those items.

\textsuperscript{15} http://networkx.lanl.gov.
4.2 Item Reputation

We mentioned in the previous section that often in online scenarios there can be more than one producer of a piece of information. Therein lies the challenge to calculating the reputation of that information: How do we deduce an item’s reputation in a way that best reflects that of its producers? In this section we describe a number of approaches to model the reputation of information produced by users in an online community. In each case, the goal is to calculate the reputation score of some user-produced item $r$ at time $t$ based on the reputation scores of the item’s producers, $\{p_1, \ldots, p_k\}$, at that point in time; see Equation 5.

$$rep(r,t) = f(rep(p_1,t),\ldots,rep(p_k,t)) .$$  \hfill (5)

For the purpose of illustration we will calculate each reputation score based on a hypothetical recommendation scenario for an item $r$ which is associated with a set of 10 producers with the following reputation scores at time $t$: $\{0.003, 0.014, 0.023, 0.052, 0.089, 0.097, 0.154, 0.297, 0.348, 0.581\}$. This includes a cross section of producers including some with low reputation scores and some with high scores. Note that in practice producer reputation scores are first normalized by the maximum producer reputation in the corresponding community to ensure a score between 0 and 1, to facilitate comparisons across subsections of online communities (subreddits on Reddit, or Twitter topics for example).

4.2.1 Median Reputation

Perhaps the simplest way to translate user reputation into item reputation is to calculate the average reputation of the item’s producers. As the distribution of reputation scores for a particular item may not always be parametric, the most useful average metric is the Median.

$$rep(r,t) = \text{median}(rep(p_1,t),\ldots,rep(p_k,t)) .$$  \hfill (6)

The advantage of this approach over a simple mean reputation is that the median statistic tends to better represent the central tendency of the set of user reputations. In the case of our hypothetical recommendation scenario the reputation of the item $r$ is 0.093 according to this median model.

4.2.2 Max Reputation

Another simple way of scoring an item based on the reputation of its producers is to take the maximum reputation value from that set. Formally, Max Reputation is calculated thus:

$$rep(r,t) = \max(rep(p_1,t),\ldots,rep(p_k,t)) .$$  \hfill (7)

Scoring pages in this way is advantageous as the reputation of an item will not be harmed if, for example, many new, not yet reputable users have selected the page. In this scenario, the reputation of item $r$ is 0.581 by this approach.
4.2.3 Harmonic Mean Reputation

Harmonic Mean is an average measure that tends towards the lower bound of a set of numbers, and thus is more conservative than arithmetic mean or median. It is calculated by finding the reciprocal of the arithmetic mean of the reciprocals. Formally, the harmonic mean of a set of user reputation scores is calculated as:

\[
\text{rep}(r, t) = \frac{k}{\sum_{i=1}^{k} \frac{1}{\text{rep}(p_i, t)}}.
\] (8)

In this case, the reputation of item \( r \) is 0.020. Harmonic mean may be a good indicator of the utility of a page in the sense that a page is only as reputable as its least reputable producer. However, rather than simply using the minimum producer reputation score available, harmonic mean permits the full range of producer reputation scores to influence the overall item reputation.

4.2.4 Hooper’s Reputation

In order to reinforce a page’s reputation according to the number of producers, keeping in mind their score, a different approach is required. Several approaches for carrying out such a task are available in past literature, for example by utilizing probability theory (Voorbraak, 1995) or by measuring the statistical distribution between samples (Bailey and Gribskov, 1998). A simpler technique is George Hooper’s Rule for Concurrent Testimony, originally proposed as a technique to calculate the credibility of human testimony (Shafer, 1986). This is applicable in our case in the sense that users who have produced a piece of information are endorsing it, in the same way that a group of witnesses might attest to the same report. Hooper gives to the report a credibility of \( 1 - (1 - c)^k \), assuming \( k \) reporters, each with a credibility of \( c \) (where \( 0 \leq c \leq 1 \)). Shafer (1986) goes on to cite an extension to Hooper’s Rule, proposed by Lambert, that calculates the probability of testimony being true if two witnesses act independently of each other. This is analogous to our reputation model where two producers of the same piece of content have different reputation scores. Because of the applicability of this approach, as well as its simplicity and effectiveness at deducing item reputation (as discussed in Section 6.3), we chose it over the other approaches mentioned.

We can extend the idea presented by Shafer (1986) to allow for any number of reporters with different values of \( c \). In an online social environment, the quality of an item can be determined by performing the calculation formalized in Equation 9, across the reputation scores of its producers. For clarity, herein we refer to this equation as “Hooper”:

\[
\text{rep}(r, t) = 1 - \prod_{i=1}^{k} (1 - c_i).
\] (9)

As per Shafer (1986), we can calculate the reputation of information \( r \) as \( 1 - (1 - 0.003)(1 - 0.014)(1 - 0.023) \) ... and so on. The reputation of this particular item \( r \) is 0.878.

If we examine each score, we can clearly see the tendencies of these item reputation models: Harmonic Mean tends towards the lower bound of the set of user reputation scores, median tends towards the centre, and Max is always on the upper bound of the set. Hooper is similarly positively inclined, with concurrent positive scores yielding a result higher than even the maximum producer reputation score. Our aim is to evaluate each of these item reputation models in terms of their effectiveness at determining item quality; this analysis is carried out in the context of the HeyStaks collaborative web search platform as described in the next section.
5 Case Study: HeyStaks

HeyStaks is an approach to collaborative web search that is designed to work with mainstream search engines such as Google, Bing, and Yahoo; so users search as normal, on their favourite search engines, but benefit from search recommendations from people they trust. The HeyStaks system has been described previously in Smyth et al (2009), where the focus was on a description of its recommendation technique. The aim of this paper is to investigate the role of a novel approach to calculating user and item reputation during recommendation, whereby the search reputation of users is allowed to influence recommendation directly. We will return to the issue of reputation in following sections, but first we present a brief review of HeyStaks in order to provide sufficient technical context for the remainder of this paper.

5.1 System Architecture

Figure 8 presents the HeyStaks architecture. There are two key components: a client-side browser toolbar/plugin and a back-end server. The toolbar, as seen in Figure 7, provides users with direct access to the HeyStaks functionality. It allows users to record and share their search experiences in repositories called “staks”. Users can create and share their own staks, according to a specific topic or community of their interest. In the instance shown in Figure 7, the user is searching for information on Cascading Style Sheets (CSS). HeyStaks recommends result pages from a set of items previously clicked on by other HeyStaks members. These recommendations appear on screen as well as the standard Google result-list. For more information on the HeyStaks toolbar and related interface, see McNally et al (2010). The toolbar also provides for the type of deep integration with mainstream search engines that HeyStaks requires. For example, the toolbar captures the routine search activities of the user (query submissions and result selections) and it also makes it possible for HeyStaks to augment the mainstream search engine interface so that, for example, HeyStaks’ recommendations can be integrated directly into a search engine’s results page.

The toolbar also manages the communication with the back-end HeyStaks server. Search activities (queries, result selections, tags, votes, shares etc.) are used by the server to update the HeyStaks stak indexes. And these stak indexes provide the primary source of recommendations so that when a user submits a query to a mainstream search engine, in a given stak context, this query is fed to the HeyStaks server in order to generate a set of recommendations based on the target stak and, possibly, other staks that the user has joined.

5.2 The Recommendation Engine

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, \( S = \{ r_1, ..., r_k \} \). Each result is also anonymously associated with a number of implicit and explicit interest indicators, based on the type of actions that users can perform on these pages, which include:

- **Selections** – a user selects a search result (whether organic (originating from normal search activity) or recommended);
- **Voting** – a user votes on a given search result or the current web page;
- **Sharing** – 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 – the user chooses to tag and/or comment on a particular result or web page.

Each of these actions can be associated with a degree of confidence that the user finds the page to be relevant for a given query. Each result page \( r^S_i \) from stak \( S \), is associated with these indicators of relevance, including the total number of times a result has been selected \( (S_1) \), the query terms \( (q_1, \ldots, q_n) \) that led to its selection, the terms contained in the snippet of the selected result \( (s_1, \ldots, s_k) \), the number of times a result has been tagged \( (T_g) \), the terms used to tag it \( (t_1, \ldots, t_m) \), the votes it has received \( (v^+, v^-) \), and the number of people it has been shared with \( (Sh) \) as indicated by Equation 10.

\[
\begin{align*}
    r^S_i &= \{ q_1 \ldots q_n, s_1 \ldots s_k, t_1 \ldots t_m, v^+, v^-, S_1, T_g, Sh \} \\
\end{align*}
\]

Importantly, this means 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\(^{16}\) index, with each result indexed under its associated query and tag terms, and this 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.

\(^{16}\) (http://lucene.apache.org)
At search time, the searcher’s query $q_T$ and current stak $S_T$ are used to generate a list of recommendations to be returned to the searcher. At search time, the source of the recommendation is not known to the searcher. For the purpose of this paper we will discuss recommendation generation from the current stak $S_T$ only, although in practice recommendations may also come from other staks that the user has joined or created.

\[ rel(q_T, r) = \sum_{\tau \in q_T} tf(\tau \in r) \times idf(\tau \in r). \] (11)

There are two key steps when it comes to generating recommendations. First, a set of recommendation candidates are retrieved from $S_T$ by querying the relevant Lucene index using the target query $q_T$. This effectively produces a list of recommendation candidate based on the overlap between the query terms and the terms used to index each recommendation (query, snippet, and tag terms). Next, these recommendation candidates are filtered and ranked. Results that do not exceed certain activity thresholds are eliminated as candidates; such as, for example, results with only a single selection or results with more negative votes than positive votes (see Smyth et al, 2009). Each remaining recommendation candidate $r$ is then ranked according to its relevance ($rel$) score, weighted with various interest indicators such as the past levels of users’ result selections, votes and shares of the candidate, at time $t$ as per Equation 11.

The relevance of a result $r$ with respect to a query $q_T$ is computed using TF-IDF, a well-known document retrieval function originating in work by Salton and McGill (1983); see Equation 11. This function 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 in-
Reputation Models for Social Recommender Systems

5.3 Reputation in HeyStaks

The above relevance model pays no attention to the source of the recommendation; i.e. the users who originally contributed the page to a stak or whose subsequent activities resulted in the page being recommended. For this reason recent research has looked at the possibility of adding a reputation component to recommendation (McNally et al, 2010, 2011). Recommendation candidates can be scored by a combination of relevance and reputation according to Equation 12, where $w$ is used to adjust the relative influence of relevance and reputation. As part of our evaluation, we attempt to find an optimal value of $w$, see Section 6. This provides the reputation of the source with influence on a page’s overall score. Pages that have been contributed by many reputable users are considered more eligible for recommendation than those that have been contributed by fewer, less reputable users.

$$\text{score}(r, q_T, t) = w \times \text{rep}(r, t) + (1 - w) \times \text{rel}(q_T, r).$$

Specifically, a collaboration event in HeyStaks occurs when a user is recommended a result page and acts upon it. In this instance, the user or users who are responsible for the page’s existence in the system become producers in the event, and the user who received the recommendation is the consumer. For more details on how the collaboration graph is built in HeyStaks, see McNally et al (2010). Although there are a range of possible actions a user can take on the result page (selection, voting, tagging, sharing) the reputation system does not discriminate according to action type. It would be possible to weight trust scores according to action type (for example, a vote up on a result page results in a greater degree of trust transferred onto producers). We leave this for future work.

The key idea in applying our collaboration-based approach to calculating reputation for HeyStaks is that reputation can be calculated by mining the indirect collaborations that occur between users as a result of their searches. For example, 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 collaboration event. The current searcher who chooses to act on the recommendation is the consumer and the original searcher (or searchers), whose earlier action on this result caused it to be added to the search stak, and ultimately recommended, is 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. Thus, we can apply any combination of user and item reputation models discussed in Section 4 to HeyStaks by integrating them into its recommendation engine according to Equation 12. Of course, the success of these models is determined by the extent to which they improve the effectiveness of the HeyStaks recommendation engine, which we analyse using a set of queries submitted to HeyStaks during the course of a live-user trial.

6 Evaluation

In previous work (Smyth et al, 2009) we have demonstrated how the standard relevance-based recommendations generated by HeyStaks can be more relevant than the top ranking

---

17 http://lucene.apache.org/core/old_versioned_docsversions/2.9.0/api/all/org/apache/lucene/search/Similarity.html
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             |

Table 1 A sample of the 20 questions presented to trial participants. The correct answers for each question are also shown.

results delivered by Google in a similar setting. In this work we wish to explore how an online social service such as HeyStaks can be enhanced by the integration of reputation information into its recommendation engine. We compare HeyStaks’ standard, relevance-based recommendation technique to an extended version of HeyStaks that also includes user and item reputation models. Earlier work has demonstrated that reputation can indeed positively influence recommendations made by HeyStaks using a single pairing of user and item model – Weighted Sum and Max (McNally et al, 2011). Now we explore different approaches to both user and item reputation in terms of their effectiveness in influencing recommendations. Specifically we consider the relative benefits (or otherwise) of different combinations of user and item reputation models.

6.1 Dataset and Methodology

Our experiment involves 58 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 effects on students. 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. A sample of the questions and their correct answers are shown in Table 1.

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.

The 58 students were randomly divided into search groups. Each group was associated with one newly created search stak, which would act as a repository for the groups' search knowledge. For this trial, we created 4 shared staks containing 5, 9, 19, and 25 users. The different sizes of shared staks provided an opportunity to examine the effectiveness of collaborative search across a range of different group sizes, which is explored in detail in
earlier work (McNally et al, 2011). In this previous work we found that stak size was not in itself a strong influencer of recommendation quality, rather the search expertise of stak members played a much more important role. As a result, the different staks can be seen as a legacy artifact of this previous experiment, and now we focus on performance across all trial participants. For this reason, in this evaluation we will focus our attention on reputation and recommendation performance averaged across these different staks rather than at the individual stak-level.

It is worth highlighting here that the setup used is such that under normal search conditions the users are likely to receive the same results for the same queries; unlike the filter bubble type of effect observed in Pariser (2011) which highlights how Google can return very different results to users for the same queries. The reason that the filter bubble does not apply here is that all users are searching in the same location, on the same network, and they were not using their personal Google accounts. This means that there was no personalization or customization of results to users.

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. All questions were attempted by at least one user during the trial, and so for the purpose of this evaluation there were no unattempted questions. Figure 9 shows the number of correct and incorrect attempts for each question, illustrating the range of user performance. Questions were answered correctly on average around 15 times, with a standard deviation of 5.3. For a more detailed analysis of overall user performance see McNally et al (2011). 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. 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 manually examined post-trial by a number of people furnished with answers to the quiz questions and its relevance with respect to the appropriate quiz question was categorized 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 10 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.

6.2 User Reputation Scores

We begin by analysing the reputation scores at the end of the 60 minute trial period that were produced by each of the four user reputation models described in Section 4. To facilitate comparisons, for each model user reputation scores are normalised by stak; i.e., for a given reputation model and stak, reputation scores are divided by the maximum user reputation score in the stak at the end of the trial. The range of normalized reputation scores across all users and all staks for each model are shown in Figure 11. In this figure, we can see the variation in scores assigned to users given by each model used.
It is clear that similar distributions were observed for the Weighted Sum (WS) and PageRank (PR) models and likewise for the HITS Hubs and Authority (Auth) models, although significant differences between the pairs of models are apparent.

What should be obvious is that all of the test users acquired at least some reputation by the end of the trial across the four user reputation models. More importantly, we can see that each of these models results in a range of different reputation values spread across the 58 users. The Weighted Sum (WS) and PageRank (PR) models, for example, generate a wide range of reputation scores (from 0.05 to approximately 1) across the users. The median reputation is about 0.4–0.45 with most users acquiring reputation scores in the range of 0.25 to about 0.6. At least some users enjoy very high reputation (up to 1) while others accumulate very low reputation scores of about 0.05. What is important here is the range of reputation values rather than the actual values. At the very least it means that users are separable in terms of reputation scores which suggests that using reputation as part of a recommendation system is at least likely to generate different types of recommendations. In the extreme case if all users ended up with the same reputation then there would be little possible benefit to be derived from incorporating a reputation model into recommendation. For the two HITS based models (Auth and Hubs) we can see that there is also a spread of reputation scores, but that these scores are less diverse that those found for WS or PR. The median reputation is considerably higher than in the WS and PR models (approximately 0.9 for Auth and Hubs) with the majority of users obtaining scores in the range 0.75–0.95. Once again the actual values are less important here than the range of values reported.

The Spearman pairwise rank correlations between user reputation scores given by each reputation model are shown in Figure 12. The chart shows that, although the scores output by HITS Hubs and HITS Authorities have similar ranges and median values, they were not assigned to the same users. This is evidenced by the relatively low Spearman Rank correlation between HITS Hubs and Authorities scores—0.44. It can also be seen from the figure that high correlations are seen between the Weighted Sum, PageRank and Authority models, with pairwise correlations between these models in the range 0.77–0.91. In contrast, the correlations between the Hubs model and the other models are much lower. It is difficult to draw precise conclusions about the Hubs correlations given the constrained nature of the user-trial, 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.

Figures 13 (a)–(d) illustrate the scores members of the 19-person stak had accumulated by the end of the trial. Here we present results from the 19-person stak as they are representative of those found in the three larger staks. Results from the 5-person stak were not as clear-cut, due to the small number of users. In the figures, users are ranked according to their Weighted Sum score in descending order, and this order is preserved across all 4 charts. The charts provide a visual example of the strong correlation between Weighted Sum, PageRank and HITS Authorities scores. However, as suggested by the correlation scores above, Hubs has scored these users differently. For example, users 14–18 have low reputation scores in all but the Hubs model, where these users were all assigned above-average reputation according to Hubs. The extent to which each of these users collaborated with their fellow stak members is illustrated in Figures 14 (a)–(d). These graphs reinforce the finding that the PageRank and Weighted Sum models are more discriminatory in determining reputable users, and thus the variation of reputation scores is greater compared to HITS Hubs and Authorities.

Although HITS Authorities and PageRank output similar scores to the Weighted Sum model, we know that the mechanism by which these scores are calculated is fundamentally different. Specifically, a user with a high PageRank score benefits from others with high scores linking in to them (i.e. collaborating with them), and a high HITS Authority user gains greater reputation when users with high Hubs scores endorse their content. Put simply, user
reputation scores are dependent on the scores of adjacent (collaborating) users, and updating one user’s reputation score has a cascading effect throughout the network. This is not true in the Weighted Sum model, which involves scoring users independently of the reputation of any other users. In a scenario such as HeyStaks using PageRank or Authorities as a reputation scoring mechanism, a user could potentially provide a stak with a single useful result that another high-reputation user could then select, thus becoming highly reputable themselves. In theory, this means the system is easier to game as reputation can be gained much more quickly. Conversely, with Weighted Sum a user must provide useful results on a more consistent basis to achieve high reputation.

Weighted Sum promotes the idea that reputation is improved over time, and a user’s reputation score according to this model is a direct function of the number of collaboration events they are involved in as producers. Thus, Weighted Sum is an inherently robust model of reputation, one in which users cannot significantly increase their reputation scores by producing less content that happens to be selected by other high reputation users as is the case with PageRank and HITS Authorities.

HITS Hubs captures a different kind of user activity compared to PageRank, HITS Authorities and Weighted Sum. Unlike those models, it measures the extent to which users...
consume information. Intuitively this goes against our primary goal – to find a metric that best expresses how effective users are at not only producing content (in the case of HeyStaks, search results), but content that others deem useful. However it is entirely possible that HITS Hubs scores can also tell us something useful. For example a user could achieve a high Hubs score by facilitating the communication of quality content in his search community.

6.3 From Reputation to Recommendation

One strong test of the reputation models in this work is the extent to which they can improve the quality of results recommended by HeyStaks. Specifically, do the reputation models help by improving the relevance of the top ranked recommendations? To test this, in our evaluation we regenerate each of the recommendation lists produced during the trial using each of the item reputation models, and based on the user reputation scores calculated at the appropriate point in time. Since we have ground-truth relevance information for all of the recommendations (relative to the quiz questions), we can evaluate the objective quality of the resulting recommendations.

For the purpose of this work 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 condition, we count how often the top-ranked result is relevant for each query and then compute a simple precision metric by dividing this count by the total number of queries considered. Thus, precision returns a value between 0 and 1 and a precision of 0.5, for example, means that 50% of top-ranked results over all queries were relevant for a given condition. This precision metric provides a clear and concise approach to compare the number of relevant versus not relevant top-ranked search results returned by each of the combinations of user and item reputation models considered in this work.

The rationale for focusing on the top recommendation instead of top \( k \) is that often during the trial HeyStaks made only a small number of recommendations. The manner in which HeyStaks operates is that it is limited to making no more than 3 recommendations as a practical way to avoid swamping the searcher with additional results. In many cases only single recommendations were made and so this was a practical approach given the sparsity of data. In a more open ended and longer-term trial it would be possible to consider the top 3 results but we leave this as a matter for future work.

In Section 4 we described four models for calculating user reputation and four different ways in which user reputation can be translated into item reputation scores. Thus, we have 16 different combinations of user and item models, and here we analyse the precision of each of these combinations when it comes to improving the quality of recommendations made by HeyStaks. Figures 15 (a)–(d) illustrate precision scores versus \( w \) (used in Equation 12 to control the relative influence of reputation) for each combination of user and item reputation model. The charts are organized according to user model, then item model. Remember that as a recommendation is made, its reputation score is calculated based on the reputation of its producers at recommendation time. Each chart shows the same value at \( w = 0 \) where reputation is not an influencing factor during the recommendation process. In each case, when \( w = 0 \), precision is 0.54. This is the baseline with which to measure the performance of each user and item reputation model combination; it represents the performance of HeyStaks using only relevance information to generate recommendations. As \( w \) increases, so too does the degree to which reputation influences recommendations, and each chart shows a similar trend. For each user reputation model there is at least one item model that brings about at least a 15% increase in precision – a precision of about 0.62 to the baseline’s 0.54. For
example, Figure 15(b) shows that using PageRank paired with the Hooper item model brings about an optimal precision score of 0.64 at $w = 0.8$, a percentage increase of 19%. This means that 19% more relevant top ranked recommendations are made over the course of the trial.

For all models, a similar trend can be seen in that precision is observed to increase initially with $w$ before decreasing as $w$ approaches 1. Thus the relevance information HeyStaks uses to rank recommendations is needed in order to optimally rank recommendations; i.e. reputation alone does not provide best performance. Peak performance occurs in a broadly similar weighting range for each technique – for example, Max paired with HITS Authorities peaks at $w=0.6$ before declining in precision; see Figure 15(d). However the exact shape of this curve differs with each model combination. For example, some combinations such as Hooper paired with Authorities or PageRank does not reach an apex until $w = 0.7$ or $w = 0.8$ before dipping.

Each user model, when paired with one or more item models, achieved a significant level of increase in precision over the HeyStaks baseline. Overall, Weighted Sum, PageRank and HITS Authorities performed particularly well when paired with the Max and Hooper item models. Equally, each of these three user models were negatively affected when combined with the Harmonic Mean item model, achieving only marginal improvement on the baseline at low values of $w$ before dipping below the baseline as $w$ increased. Weighted Sum performed well when combined with Median with a top precision of 0.62 before descending back towards the baseline as $w$ approaches 1. Although the trend is similar for HITS Author-
ities and PageRank, the top precision score was much lower when Median was used. This is indicative of the general trend showing Weighted Sum as the top performing user reputation model. Overall, HITS Hubs performed the least well of the user reputation models, achieving a maximum precision of 0.62 (combined with both Hooper and Max), the lowest across user models.

Examining the performance of item models, Hooper and Max consistently achieved the best precision across all user models. In the case of HITS Authorities for example, Hooper performed best by a considerable margin, with an optimum precision of 0.65. When combined with Weighted Sum, Hooper achieves the top precision observed in the experiment: At both \( w = 0.4 \) and \( w = 0.8 \) the combination achieved a precision score of 0.66 – a 22\% improvement.

Not all item reputation models performed well. Median was an inconsistent performer, yielding a small degree of improvement when paired with PageRank, HITS Hubs and Weighted Sum, but little or no performance improvement in the case of HITS Authorities. The Harmonic Mean reputation model on the whole performed poorly regardless of the user reputation model it was paired with. In fact in many cases this item model yielded smaller precision values than the HeyStaks baseline. For example, a combination of Harmonic Mean and PageRank leads to a precision of only 0.47 when \( w = 0.9 \). So not only would users receive relevant recommendations less frequently than if standard HeyStaks was used (according to the baseline precision of 0.54), they would receive fewer relevant recommendations than not relevant ones.

Overall Figures 15 (a)–(d) show that including reputation information in the recommendation process can bring about considerable improvement in the system’s capacity to deliver relevant recommendations to its users. In fact, when using Weighted Sum paired with Hooper’s Reputation model we see up to a 22\% percentage increase in precision over the HeyStaks baseline, a testament to the success of pairing reputation with the default HeyStaks’ relevance-based scoring mechanism.

6.4 Analysis of Recommendations

The precision scores reported demonstrate different levels of benefit overall across the different approaches considered. But it is interesting to consider the source of these benefits. For example, do these benefits arise because of improvements in the recommendations associated with just one or two questions, or are they evident across a broader set of questions?

To explore this we analyzed the data to identify those questions that enjoyed an improvement in precision and found that in the main, and depending on the particular approach used, an improvement was found across a range of questions. For example, in the case of our best performing combination – Weighted Sum paired with Hooper – precision was achieved across queries matched to 9 of the 20 questions posed; in other words for this combination 45\% of the questions enjoyed recommendations that represented an improvement over those provided by HeyStaks alone. Figure 16 summarizes the results across the other combinations, using the optimal value of \( w \) in each case. We have also overlaid the precision achieved by each approach. A correlation can be seen between precision and number of questions over which an improvement was observed \( (r = 0.74) \).

Why did some questions enjoy benefits while others did not? It is likely that this is due to a combination of factors. Clearly the approach to reputation modeling is a factor in that different combinations of techniques deliver different overall precision improvement and across different questions. Question difficulty is also likely to play a role here in that there
is more opportunity for improvement, and therefore an increase in precision, for more challenging questions. It is also likely impacted by the random ordering of questions presented to users in the trial.

6.5 Summary Analysis

To facilitate a more direct comparison of the performance of user and item reputation models, Figure 17 shows, for each item model, the median precision scores over $w$ for each user model. Here, it is clear that the best performing item reputation model is Hooper, which outperforms all other item models irrespective of the particular user model employed. Further, it can be seen that the Weighted Sum user model outperforms the other user models (except when used in conjunction with Harmonic Mean). Overall, the best user-item model combination is Weighted Sum and Hooper, achieving a precision score of 0.66 compared to 0.65 (Figure 15) for the next best combination of Authorities and Hooper, and similar if not significantly greater performance improvements over the remaining user-item model combinations.

A Kruskal-Wallis one way analysis of variance test on precision scores across weights indicates there are statistically significant differences between the performance of the reputation model combinations at the 0.01 level. Using the Tukey-Kramer Method we examined pairwise differences between model combinations at the 0.01 level. A total of 12 pairwise comparisons yielded significant results. We observed that Weighted Sum, PageRank and Hubs, when paired with the Hooper item reputation model, were our top performers. These three combinations performed significantly better than four of the other user and item reputation model combinations: Weighted Sum with Harmonic Mean, PageRank with Harmonic Mean, and Authorities with Median and Harmonic Mean. No significant differences were observed between any of the other pairs of reputation model combinations. These findings confirm the strong performance achieved by three of the four user models and, in particular, the Hooper item reputation model.
6.6 Discussion

The results of this user trial clearly show the benefit that comes from integrating reputation information into the HeyStaks recommendation engine. We present our key findings in comparing combinations of user and item models to the HeyStaks baseline over the set of user queries obtained from the user trial:

1. Generally speaking a number of combinations of user and item reputation models are capable of delivering improvements in precision with respect to the HeyStaks baseline.
2. The precision scores recorded increase with respect to $w$ to a point and typically apex in the $w = 0.6 - 0.8$ range.
3. Weighted Sum outperforms other user reputation models, regardless of item reputation model.
4. Hooper is the best performing item reputation model, delivering an average precision increase of 15% across all user reputation models, relative to standard HeyStaks.

The fact that Hooper and Max are significantly more effective at ranking pages than Median and Harmonic Mean is not surprising. Given a set of producer reputation scores for a candidate recommendation, Hooper and Max tend towards the upper bound of the distribution of scores, whereas median and harmonic mean provide a much more conservative reputation measure. For example, imagine a scenario where two very reputable users produce a result page, then two new users with low reputation select this page and become producers for future recommendations. In this case, it stands to reason that we would want to reward that page with a high score given its two more reputable producers rather than penalize it simply because two new users are also producers. However, Hooper’s high performance may be peculiar to this trial due to the nature of the task assigned to participants: as the search task was the same for everyone, users tended to converge on the same pages. This meant that the number of producers of a page grew as the trial wore on. As such, the variation of producer reputation scores for a given page became greater. Hooper and Max still rewarded those pages with high reputation whereas Harmonic Mean penalized them, tending towards the lower bound of the set of scores. Anecdotally we have found that the context-aware nature of HeyStaks results in users converging on pages in staks, even in
a real-world setting, and thus we consider that Hooper or Max are the most suitable item reputation models going forward.

Differences between the user reputation models can be explained by examining how each user model scores users after each collaboration event. For example, one user may have contributed a single page to a stack that was recommended to and selected by a high-reputation consumer. In the case of Weighted Sum the reputation of this consumer is irrelevant from the point of view of updating the producer’s reputation. In contrast for PageRank and both HITS Hubs and Authorities, the reputation of the consumer is taken into consideration when the link between consumer and producer is analyzed. Put simply, a previously low reputation user can gain a high degree of reputation by virtue of one high reputation user selecting that one recommendation. If we are scoring users based on their level of engagement with users on a collaborative platform and to prevent gaming, the degree with which they collaborate with other users should be the primary consideration. Since these criteria are best captured by the Weighted Sum model, we believe that this model is the most suitable of those considered.

Interestingly Hubs was a good performer, in fact Figure 17 shows Hubs outperformed both HITS Authorities and PageRank when coupled with Max, Median and Harmonic Mean item models. In a sense, this finding is counter-intuitive and highlights an interesting property of the HITS algorithm in this context. One might expect, for example, that the Authorities model would outperform Hubs, given that Authorities 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 manner in which the collaboration graph is constructed (Section 3.2), once a user has consumed (selected) a recommended result, then that user is also considered to be a producer of the result in question if it is subsequently recommended by HeyStaks and then selected by other users in the future. Thus, good consumers — who select recommended results from many good producers (i.e. producers with high Authority scores) — serve a “filter” for a broad base of quality search knowledge. Therefore re-ranking default HeyStaks recommendations using reputation scores from the Hubs model leads to the good recommendation performance observed above. However, as stated previously, a significant limitation of the Hubs model is that it is vulnerable to a simple form of gaming; for example, by adding many results to stacks, users can accrue high reputation according to the Hubs model which is clearly not a desirable property from a robustness perspective.

6.7 Limitations of the Experiment

This article speaks to the benefit of utilizing reputation information to positively influence content presented by social recommender systems. We have shown this in the context of a live-user trial of the social search utility HeyStaks. Using reputation information gained by employing various user and item reputation models, we improved on the frequency at which HeyStaks makes relevant recommendations. We have presented generic models of user and item reputation, however in this study we have only showed how the models can be used in one of many possible platforms. The next important step is proving the utility of this approach in other domains. Once this is achieved, a key question to answer is can reputation scores from multiple domains be aggregated? We view such a question as an important one
to consider not just to augment this work, but as an important one for all future work in the area of online reputation.

Of course we acknowledge that the trial described in this paper has its limitations and that our results must be viewed in the context of 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 (2009). Nevertheless, our closed fact-finding search task does provide useful insights and facilitates a thorough evaluation with respect to an independent model of result relevance, in which the absolute relevance of individual results is known.

Finally, we demonstrated the benefits of our proposed approach 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 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. Given that we have a ground-truth for relevance of recommended result pages, we can effectively replay queries that were entered by live-users, and this ground-truth allows us to measure relative recommendation efficacy with great confidence.

7 Conclusions

The ability to assess the reputation of online users is an important research area especially as we come to increasingly rely on social networks and connections for a wide range of tasks, from information finding to e-commerce and communication. The central contribution of this article is a generalized approach to calculating user and item reputation based on the collaboration (implicit or explicit, direct or indirect) that naturally occurs between users in a variety of online tasks and scenarios. Early work on computational reputation systems (Resnick et al, 2000; Shneiderman, 2000) asserts that users’ past performance in online, interpersonal relationships can provide an indication of the extent to which others in their community trust them, and thus their overall reputation within that community. The reputation models put forward in this article are based on that assertion, leveraging information gained from examining past user collaborations to positively influence interactions in the future. This particular approach calculates reputation by using a collaboration graph as a harness. Past work has examined how a graph can be built using trust information explicitly
and directly given by users (Massa and Bhattacharjee, 2004; Massa and Avesani, 2007; Golbeck, 2006; Kuter and Golbeck, 2010). This article explores how a graph can be constructed using collaboration information that may be explicit or implicit, and direct or indirect. From this graph the reputation of the users within it can be calculated. We have shown how this approach can be developed in a social search context, proposed a variety of user and item reputation techniques, and demonstrated its overall effectiveness in the context of a live-user trial.

In this article we have explored different ways to translate user reputation scores into an overall page reputation/relevance score. We have described the results of a comparative evaluation based on real-user data, albeit in a constrained search setting, which highlights the ability of these techniques to improve overall recommendation quality, when combined with the relevance-based recommendation ranking metrics that are currently used by HeyStaks. For example, many of the page reputation models can improve precision in delivering relevant recommendations (compared to the standard HeyStaks benchmark) by over 15%. Moreover, we have found that by combining our Weighted Sum user reputation model with an enforcement-based item scoring mechanism based on Hooper’s rule for Concurrent Testimony (Shafer, 1986), relative performance improvements of up to 22% are delivered. 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 online social platforms and refining the combination of reputation and content utility.

Our focus in this work was on the role of reputation during the recommendation process, in order to maximise the relevance of the community recommendations made by HeyStaks. But this is just one use of reputation in a system such as HeyStaks. For example, in many social systems there is the risk that malicious users will attempt to manipulate the outcome of social processes; see relevant work in recommender systems research (Lam and Riedl, 2004; Mobasher et al, 2007; O’Mahony et al, 2002). In HeyStaks, for example, it is possible for malicious users to flood search staks with irrelevant or self-interested results, which could impact recommendation quality. By using reputation to mediate recommendation it will be possible to guard against this; these malicious users will have low reputation scores (assuming their contributions are rarely acted on by other users) and as such, their contributions will be unlikely to appear in future recommendation sessions. Furthermore, reputation can be exposed to users of systems like HeyStaks as an important social signal. For example, although HeyStaks’ recommendations are anonymous (so users do not know the source of result recommendations at search time), it may make sense to explain recommendations with reference to the reputation of producers in the future.

References


Bakshy E, Hofman JM, Mason WA, Watts DJ (2011) Everyone’s an Influencer: Quantifying Influence on Twitter. In: Proceedings of the fourth ACM International Conference on Web Search and Data Mining (WSDM ’11), Hong Kong, China, ACM, New York, pp 65–74


Author Biographies

(1) Kevin McNally, B.A.

University College Dublin, School of Computer Science and Informatics, Belfield, Dublin, Ireland

Kevin is a PhD candidate currently working under the supervision of Professor Barry Smyth and Dr Michael O’Mahony in the School of Computer Science and Informatics at University College Dublin, Ireland. Kevin received his B.A. in Computer Science from University College Dublin in 2009. Kevin’s main research interests include Online Trust and Reputation Modeling, Information Retrieval, Recommender Systems and Social Network Analysis.

(2) Dr Michael O’Mahony, B.A. B.A.I., H.Dip. Computer Science, Ph.D.

University College Dublin, School of Computer Science and Informatics, Belfield, Dublin, Ireland

Michael is a Lecturer in the School of Computer Science and Informatics at University College Dublin, Ireland. Michael received his B.A. B.A.I. degree in Mechanical Engineering from the University of Dublin, Trinity College, Ireland and his Ph.D. from University College Dublin, Ireland. He has 10 years of research experience in the areas of Recommender Systems, Reputation Systems, Personalisation, Web Search and User-Generated Content Analysis. He has published over 60 peer-reviewed journal and conference papers in his areas of expertise.

(3) Prof. Barry Smyth

University College Dublin, School of Computer Science and Informatics, Belfield, Dublin, Ireland

Prof. Barry Smyth holds the Digital Chair of Computer Science in University College Dublin. He is the Director of CLARITY: The Centre for Sensor Web Technologies, a Science Foundation Ireland-funded Centre for Science and Engineering Technologies. His research covers a broad set of topics within Artificial Intelligence including Case-Based Reasoning, Machine Learning, User Modeling and Planning with particular focus on so-called Personalisation techniques.
Part III

GENERALISING THE APPROACH

The previous two sections (and five papers) have shown how reputation can be modeled in a social system, and how this information can be leveraged to improve the quality of recommendations made in a social recommender system. We have focused on Social Search and HeyStaks in particular but of course we have developed our approach to be applicable across other social services and applications.

With this in mind, we present the sixth and final paper contained in this thesis. This paper describes work carried out on adapting the model to integrate in a second domain: Social Question and Answering services.
PAPER 6: A MODEL OF COLLABORATION-BASED REPUTATION FOR THE SOCIAL WEB

Kevin McNally, Michael P. O’Mahony and Barry Smyth
Published in ICWSM ’13: Proceedings of the 7th AAAI Conference on Weblogs and Social Media (2013)
A Model of Collaboration-based Reputation for the Social Web

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

Abstract

In this paper we describe a generic approach to modeling user reputation in online social platforms based on an underlying model of collaboration. This distinguishes our approach from more conventional reputation models which are often based around ad-hoc activity metrics. We evaluate our model with respect to a conventional reputation model used by 3 social Q&A websites, each based on a different topical domain.

Introduction

The social web reflects an important paradigm shift in the nature of our online transactions. We increasingly rely on the views and opinions of others to mediate these transactions and as such the reliability of these users becomes an important indicator of quality. Thus, concepts like trust and reputation have become increasingly important in the context of today’s social web. It is not surprising that there has been considerable recent research on various ways to measure, quantify, and evaluate the trustworthiness and reputation of users as they participate in a diverse array of online interactions (Resnick and Zeckhauser 2002; Cheng and Vassileva 2005; Recuero, Araujo, and Zago 2011).

In this paper, when we refer to trust we are referring to an asymmetric relationship between two users: user $A$ can trust in user $B$ to a lesser or greater extent. Trust refers to the feedback provided by one user in the context of some interaction with another user. For instance, on eBay a buyer can rate a seller and this corresponds to a degree of trust that the buyer has in the seller. When we refer to reputation we are referring to a measure that is associated with an individual user, generally as some function of the individual trust scores that have been assigned to this user. For example, an eBay seller might have an average rating of 90%, which corresponds to their reputation in the sense that it is a measure of their trustworthiness across multiple transactions. The idea of distinguishing online trust from reputation has been explored by (Mui, Mohtashemi, and Halberstadt 2002).

In previous work we have proposed a collaboration-based model of reputation for the social search utility HeyStaks (McNally et al. 2011). The key contribution of this paper is the examination of the idea that such a model generalizes to other domains, in this case, Social Q&A. Recent work indicates that a big motivation behind applying reputation systems is to incentivize users of a platform to engage online with others. This can be aided by users trusting the system (Joinson 2008), but also by rewarding users with positive feedback (Cheng and Vassileva 2005) or with improved social standing (Recuero, Araujo, and Zago 2011). A successful reputation system can not only distinguish trustworthy users from untrustworthy ones, but also determine the quality of the resources users provide. Many of these systems calculate trust or reputation by building a network based on some explicitly or implicitly gained information in a specific context. To date, work has focused largely on developing trust/reputation models in a specific setting or context. For example, (Hong, Yang, and Davison 2009) examined modeling reputation on social Q&A sites by leveraging the links that exist between users based on “best answer” feedback.

In this paper we intend on extending a generic model of collaboration-based reputation to the same domain. We take a principled approach to calculating reputation. In particular, we consider the following criteria in our model:

1. **What is the structure of the model?** In every case we can model an online community as a graph that makes explicit the interactions between producers and consumers of information. For example, in a social Q&A setting, which we will examine in detail in this paper, interactions between users can be viewed as users voting on each other’s produced answers.

2. **What are the fundamental units of collaboration between users?** The answer to this question is dependent on the domain being examined. In social Q&A, collaboration can be viewed as one user voting on another user’s answer to a question posed by a member of the community, resulting in an edge being drawn in the graph between the pair of users. We refer to such a singular instance of interaction as a collaboration event.

3. **How do we aggregate units of collaboration to calculate reputation?** How can we leverage the level of collaboration occurring between members of an online community to deduce their reputation? This can be achieved using any number of node aggregation techniques.
A Computational Model of User Reputation

Our model of reputation is based on naturally occurring collaboration events that are frequent in many different social web settings. These events can be associated with trust scores to reflect the quality of interaction between collaborating users. Collections of such events naturally form a collaboration graph, and trust scores that flow between pairs of users can be aggregated to evaluate the reputation of individual users. It has been shown in previous work that user reputation information can be utilized to positively influence recommendations made by the social search utility HeyStaks (McNally, O’Mahony, and Smyth 2011; McNally et al. 2011). Here we show how this model can be generalized to other domains, focusing on calculating user reputation as a means of discovering the quality of users’ answers on the Stack Exchange network. We evaluate this model by measuring its correlation to a ground-truth metric, and comparing our own reputation scoring metric—”WeightedSum” (WS)—to that of standard PageRank (PR) (Brin and Page 1998), as outlined below. For more details on these metrics and how they are employed in a reputation scenario, see (McNally, O’Mahony, and Smyth 2011). Finally, we compare both these approaches to calculating reputation with scores calculated by Stack Exchange (SE), using their own activity-based reputation framework.

Reputation in Social Q&A Sites

The Stack Exchange network is a popular group of social Q&A sites. There are currently 83 different topical Stack Exchange websites, hosting almost 2 million users. Some 3.8 million questions have been posed, eliciting 7.7 million answers. Users are permitted to post questions that can be answered by other users in the community. Each answer given can be voted up or down by others and the questioner can choose to highlight a single answer as correct; indicating the question has been answered satisfactorily or that answer was the best answer provided.

Collaboration-based Reputation on Stack Exchange

The reputation model proposed can be applied to Stack Exchange data, assuming that a suitable collaboration event can be defined. In this particular scenario we have chosen to model collaboration events as instances of question-answers with individual question-answer pairs scored according to the relative proportion of votes received.

On Stack Exchange sites, users can give feedback on questions and answers by positively or negatively voting on them, and these votes are aggregated by summation. For the purpose of our collaboration events we only include answers that have received a positive aggregate score; an edge is drawn between two users only if the user’s answer has received more positive than negative votes from their community. In this case of a given set of question-answer pairs it leads to the partial collaboration graph shown in Figure 1. In turn each collaboration event is scored based on the proportion of votes that the corresponding answer has received.

Using this voting data, we define the following collaboration event. For each answer that received an aggregate vote greater than 0, the associated producer (i.e. the user who provided the answer) receives a trust score in proportion to the vote received. For example, suppose a question is posed on the site and receives five answers, each of which receive an aggregate vote score of +5, +4, +2, -1, -2, respectively. Let $u_{a1}, u_{a2}, \ldots, u_{a5}$ denote the producers of these five answers and let $u_q$ denote the consumer who posted the question. After eliminating answers with negative net votes, the unit of trust scores are divided between producers $u_{a1}, u_{a2}$ and $u_{a3}$ in the proportions $\frac{5}{11}, \frac{4}{11}$ and $\frac{2}{11}$ respectively; see the edge weights in Figure 1.

Once the collaboration graph has been created, by adding edges for all of the collaboration events in a given Stack Exchange dataset/domain, the reputation of each user can be calculated according to WS and PR. Briefly, WS is a count of all weights on incident edges into a given (producer) node; note a user who is part of the graph but has no incident in-links will not receive a score (see (McNally, O’Mahony, and Smyth 2011) for details). This is unlike PR, where all connected users receive a small default score.

Once the reputation of users has been calculated, it can then be leveraged for a variety of purposes; for example, to re-rank answers to questions based on the answers’ reputation or to provide a means to automatically route questions to users with high reputation in the domain. In this paper, we focus on computing user reputation and leave such applications of reputation to future work.

Evaluation

The current reputation model in Stack Exchange is a typical conventional model based on an aggregation of various user activities on the site, and as such it does not represent a ‘pure’ model of user reputation. Further, it is a domain-specific approach, requiring a fine-grained weighting of the significance of the various activities users can perform on the site. In this evaluation we compare our collaboration-based reputation model, which assumes little domain knowledge, to this conventional approach.

Dataset and Methodology

For this evaluation we selected three Stack Exchange sites, crawling each site to download a complete picture of user activity. These datasets are summarized in Table 1.

---

1. http://www.stackexchange.com
To conduct our experiment we only considered questions that received at least one answer and where answers have received at least one vote from the community. The smallest of the three datasets examined is from the Seasoned Advice website, dealing with topics on cooking. A similarly sized dataset is taken from the Electrical Engineering website, and the third and largest dataset is from the Mathematics website. We chose the three datasets based primarily on the nature of questions asked on each site. For example, the Cooking dataset provided us with answers to questions that can be more qualitatively assessed by the community. Conversely, answers given on the Mathematics website tend to be either right or wrong, although answers can be voted on with the manner of delivery in mind. The Electrical Engineering dataset provides both answers that can be interpreted as right or wrong, and answers that can be assessed qualitatively.

We compare the WS and PR variations of our collaboration-based model by building a collaboration graph for each dataset as previously described. Figure 2 shows histograms which illustrate the distribution of reputation scores across each of these models and datasets. Distributions of user reputation according to the conventional SE model are also shown. For clarity, we have normalized each user’s reputation score by the maximum score found in each dataset, according to the individual reputation model presented. All charts indicate the long-tailed nature of user reputation. Each reputation model tends to distribute reputation scores in a similar way for each dataset, although proportionately more users receive a score between 0.05 and 0.1 in the Seasoned Advice dataset compared to the other two datasets.

In order to evaluate these three reputation models we establish a ground-truth as the basis for comparison. In the Stack Exchange network each questioner has an opportunity to mark a single answer as correct. This is clearly a strong indicator of answer quality and it is reasonable to consider users who are associated with many correct answers to be more reputable than users who are associated with fewer (or no) correct answers. Table 1 shows the number of users who have answered at least one question correctly according to the questioner. As such, for the purpose of this evaluation, we will use the number of correct answers for a user as a ground-truth for their reputation. And by correlating the reputation scores from the three models with respect to these ground-truth scores we can analyse the performance of each model; under the assumption that higher correlations are to be favoured because they indicate a given reputation model as a stronger predictor of the ground-truth.

### Correlation Analysis

Ultimately the true test of our evaluation models is the extent to which reputation correlates with our evaluation ground-truth; in this case the number of correct answers provided by a user. Figures 3(a)–3(c) show the correlations between each reputation model and the ground-truth for different sized groups of users in descending order of reputation. In each chart the maximum number of users corresponds to the number of users for whom ground-truth information is available. For example, in Figure 3(a) we can see that for the top-50 most reputable users in the Seasoned Advice dataset, WS enjoys a correlation coefficient of 0.95 with the ground-truth.

Overall we can see that the correlation achieved by the WS model tends to degrade as we consider larger sets of users with decreasing reputation values. This is to be expected. As we verge towards users with lower levels of activity a small difference between two users’ reputation scores becomes more pronounced, having a greater effect on their rank, and thus the correlation. In contrast, the correlation of the PR model tends to increase as we consider larger sets of users with decreasing reputation values. In this regard, the assignment of default reputation scores to users by the PR model has a positive effect on the ranking of users with small

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SA</th>
<th>EE</th>
<th>Math</th>
</tr>
</thead>
<tbody>
<tr>
<td># Users</td>
<td>7,555</td>
<td>9,692</td>
<td>22,342</td>
</tr>
<tr>
<td># Questions asked</td>
<td>5,133</td>
<td>7,783</td>
<td>35,251</td>
</tr>
<tr>
<td># Answers</td>
<td>15,087</td>
<td>18,900</td>
<td>4,863</td>
</tr>
<tr>
<td># Correct Answers</td>
<td>7,456</td>
<td>4,363</td>
<td>2,979</td>
</tr>
<tr>
<td># Votes on Answers</td>
<td>46,947</td>
<td>20,614</td>
<td>2,608,826</td>
</tr>
<tr>
<td>Users with ≥1 Questions Answered Correctly</td>
<td>779</td>
<td>604</td>
<td>1,842</td>
</tr>
</tbody>
</table>

Table 1: Summary statistics for three Stack Exchange datasets: Seasoned Advice (SA), Electrical Engineering (EE) and Mathematics (Math).
levels of activity, leading to higher correlations in our experiments. In addition, the effect of propagating scores featured in the PR model is likely a primary cause of the degradation in rank correlations between the top 50 or 100 users and the ground-truth. For example, the top ranked user in the Electrical Engineering dataset according to PR is ranked second by the other two models and, indeed according to our ground-truth. Similarly in both the Mathematics and Seasoned Advice datasets, PR is good at identifying the top 50 users in accordance with the ground-truth, however their ranks are considerably different. Another feature of PR that may be having a negative effect is the fact that an individual’s reputation influences that of adjacent users. This idea has been explored in (Hong, Yang, and Davison 2009).

In Figures 3(a)–3(c) the WS model consistently outperforms SE, which in turn outperforms PR. We then conclude that our WS model performs very well with respect to the ground-truth, delivering effective reputation estimates per user and importantly without the need for fine-grained tuning or deep domain knowledge.

Conclusions

The key contribution of this work is the presentation of a principled approach to modeling user reputation in online social platforms. Collaboration between users can be leveraged to measure the reputation of users in potentially any online collaborative environment. We present this approach to calculating reputation as an alternative to more traditional activity-based models. In particular we have compared two variations of our model to the current Stack Exchange reputation model using data drawn from three diverse Stack Exchange Q&A domains. In each case we examined the correlation between user reputation and the number of correct answers provided by users as a ground-truth to demonstrate superior performance for our WS variation when compared to the SE and PR alternatives. This analysis is not without its limitations, particularly given that only one of many possible interaction scenarios was examined. In the future it would be useful to analyze a reputation model against other types of interaction and, indeed, examine the effect on reputation of building the collaboration graph using different kinds of activity.

Acknowledgments

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

References


The articles presented in this thesis aim to explore the idea of how we can reliably calculate the reputation of online collaborative users, and utilize reputation information to enhance the quality of recommendations made by a social recommender system. Below we provide an outline of each paper’s key contributions. In Section 8.3 we discuss possible directions this work could take in the future, as well as questions posed by the findings of the work.

8.1 SUMMARY OF CONTRIBUTIONS

8.1.1 A Computational Model of Reputation

The articles contained in this thesis introduce and give a detailed description of a principled model of reputation that is based on underlying collaborations that occur on online social systems.

- **The Fundamental Units of Collaboration** The fundamental unit of collaboration was defined, inferred from individual collaborating events occurring between users. Each collaboration event involves one or more *producers* supplying some piece of content to their community, and a single user – the *consumer* – then consuming it. As a result, a unit of trust is conferred to the producer(s). This collaboration can be explicit or implicit, and direct or indirect (that is, direct collaboration, or collaboration mediated by an item) depending on the platform and the nature of this interaction. In a social search setting for example, it was found that a collaboration occurs when a user acts on a search result that has been provided by another user in their community.

- **A Graph-based Model of Collaboration** As users interact with a system over time, they participate in a number of collaboration events, and can assume the role of either producer or consumer in multiple collaboration scenarios. From this information we can model an online community as a graph that illustrates how users produce and consume each other’s content. Each node
in the graph represents an individual user, and the links represent instances of collaboration.

Using this model, we created collaboration graphs for a Social Search utility using both real-world and live-user trial data, and for a number of Social Q&A platforms. From each graph we could make inferences about the reputation not only of each user in the network, but measure the reputation of the content the community was producing.

- **Modeling User Reputation** Reputation can be calculated for each user of a social platform by examination of this underlying collaboration graph. Individual trust scores conferred onto users as a consequence of individual collaboration events can be aggregated to calculate individuals’ reputation scores. In order to aggregate the scores at the nodes contained in the graph, we introduced the scoring mechanism *WeightedSum*, which for each node is a weighted count of its incident edges. That is, the mechanism counts the number of times a user assumes the role of producer in collaboration events. We also calculated reputation using the well-known link analysis approaches PageRank [56] and HITS [29]. This approach to calculating user reputation was evaluated in terms of how effectively it indicates the quality of activity users partake in. Using real-world and live-user trial data, reputation was calculated for a number of users of a social search system. Various activity and search performance metrics were recorded for the same users that indicate search expertise, and reputation was found to correlate well with these metrics.

8.1.2 **From User Reputation to Item Reputation for Recommendation**

A key contribution of this thesis is to show how reputation information can be used to positively influence recommendations made in a social recommender system. In order to affect how content is recommended to users, we introduced the concept of translating user reputation to item reputation. We were then able to integrate item reputation scores as a consideration for the recommendation engine of a social search platform.

We described five mechanisms to calculate the reputation of an item based on that of its producing users: *Maximum, median, harmonic mean, root mean squared* and a reinforcement technique we refer to as *Hooper’s Reputation* based on Hooper’s Rule of Concurrent Testimony [61].
Every combination of each user and item reputation scoring mechanism was evaluated in terms of their efficacy in improving recommendation quality. Using live-user trial data, a set of recommendations could be re-ranked using each approach, and it was found that including reputation in the recommendation process improved the relevance of top-ranked recommendations made to users. The top performing combination was our own WeightedSum paired with Hooper’s Reputation, which was found to yield a 22% improvement in recommendation precision over HeyStaks’ standard recommendation engine.

8.1.3 Generalising the Approach

The first five articles contained in this thesis discuss a model of collaboration-based reputation in the context of social search. The sixth and final article shows how the same approach can be transferred to the online Social Q&A domain, with the motivation of demonstrating how the approach can generalise. The key to application of the model to a new domain is to identify the collaboration events that occur on the platform. In the case of Social Q&A, a collaboration event occurs when a user provides an answer that is up-voted by another member of their community. This approach was applied and evaluated using real-world data from three Social Q&A datasets.

Scores generated by the WeightedSum and PageRank user scoring mechanisms were found to correlate closely with the number of questions they have answered correctly – a ground-truth metric that demonstrates user authority. In fact, WeightedSum scores correlated more closely with the ground-truth metric than the platform’s own bespoke, activity-based reputation scoring system.

8.2 Summary of Limitations

The six articles and conference papers included in this thesis all describe key findings from experiments performed which validate the core hypothesis that reputation information can positively influence recommendations generated by a recommender system in an online social context. A number of limitations relating to these experiments were also described; what follows is a summary of those limitations:

- The number of trial-users in both HeyStaks experiments was quite small (26 and 64). The first trial was preliminary in nature, utilizing the small user-base
whose activity was available for examination. The second user-trial involved closer examination of a set of users of a similar size. In both cases, this was a feasibility issue: At the time of each experiment, it was not possible to test HeyStaks on the kind of scale that would be normal for mainstream search engines, either technically or because of the nature of its user-base. This limitation is particularly acknowledged in Paper 5.

- The experiment described in Papers 2–5 was closed in nature. In order to measure the effect of reputation information influencing HeyStaks’ recommendation process, a ground-truth for result relevance was required. In order to obtain this ground-truth, the trial was closed in nature, effectively limiting the number of results that had to be tested for relevance to the quiz presented to trial users.

- This experiment involved a single type of search task, rather than depicting a more general search engine use-case. Answering quiz questions is an informational search task, and thus represents one of the three major search tasks as described in [8]. Testing its ability to positively influence recommendations for other query types would be a key matter for future work.

- The reputation model was evaluated in an offline manner throughout the experiments. Specifically, in each case a set of activity data was recorded, then reputation applied as part of a post-hoc analysis. This allowed for the evaluation of many different aggregation techniques to attribute reputation scores to both users and items, as part of a single experiment. An obvious next step for this work would be to develop and deploy the reputation model as part of a live system, and develop experiments around the system in order to evaluate it.

8.3 Future Directions and Open Questions

8.3.1 Beyond Social Search

The work described in the final article of this thesis demonstrates how collaboration-based reputation can be applied in a Social Q&A setting. Although this is an interesting finding in and of itself, there is work to be done in order to understand the extent of its utility in that domain. Recent work has been carried out with the aim
of evaluating the quality of information provided on Social Q&A sites, for example in determining whether an answered question is worthy of archival for re-use [21]. Morris et al. [44] consider the benefits of injecting Social Q&A discussions originating from a user’s social network into results returned by standard search engines. This would come with its own set of challenges, for example measuring the utility of answers provided for the purposes of ranking. Examining related work informs possible next steps in demonstrating how reputation can be evaluated and utilised in the domain. For example, can user reputation be used to re-rank answers provided to users’ questions? Similarly, reputation information could be employed to route questions to individuals that are known authorities on a topic.

The above innovation would work on the basis that a system knows which users are reputable according to specific topics. The two domains explored in this thesis allow for that to happen independently of the reputation system: A key concept of HeyStaks is the Search Stak, which can be used to house topic-specific search knowledge. Stack Exchange’s websites are topic-specific. In HeyStaks, it would be simple to employ a system whereby a person’s reputation only extends as far as a single Stak. Similarly, a person may be a high-reputation user in Stack Exchange’s Mathematics website, but that has no bearing on their reputation in the “Seasoned Advice” cookery website, should they choose to become a member. Many collaborative platforms do not distinguish user behaviour based on topic. For example, how difficult would it be to gain utility from the reputation system if it was employed on the Quora¹ Social Q&A site, which does not provide for the same topic granularity? Might we provide Wikipedia users with have a separate reputation score for each article they contribute to? Such questions are vital to understanding the limitations of the model, or indeed any principled reputation model, and its generalizability.

Social Q&A and Social Search are two of many areas in the online social space. The approach to calculating user and item reputation could be tested on any platform that allows users to collaborate. In our work we show how reputation information can be used to enhance recommendation quality in a social recommender system. A prominent matter for future work is testing this aspect of the approach in a second domain, whether in Social Q&A or indeed on a new platform.

¹ http://www.quora.com
8.3.2 Integration and Scalability

Testing the model on multiple domains may very well result in dealing with users and content on a much larger scale. Evaluation of the reputation model described in this thesis was carried out primarily using live-user trial data. A logical next step is to explore user reputation on a live system over a longer period of time, and involving a larger group of users. This brings to light potential challenges around building a collaboration graph across users at a greater scale. For example, how would the algorithm described in Paper 1 perform on a community containing millions of members? In both research and industry, similar questions have been considered. For example, the work presented in [38] looks at link-analysis processing on large-scale web graphs (of the order of billions of nodes and trillions of edges). This work has resulted in the Giraph Open-source project², which is used by Facebook to perform analysis of their social network at scale. Often systems that analyse social networks at scale tend to involve iterative techniques as part of an offline process. In a social recommender system context, it may be necessary to calculate reputation in real-time as collaborations occur. An advantage of the WeightedSum scoring technique compared to, for example, PageRank, is that if a new collaboration event occurs within a network, only the reputation of the users involved in the collaboration is updated. As such, updating reputation scores is much less computationally expensive, and thus easier to scale.

Deploying a reputation model in a live system brings to light a large number of interesting research questions that might reveal positive properties and strengthen its utility to that system. For example, sites such as the Social Q&A network examined in this thesis publicly displays users’ reputation scores as a way to incentivise engagement and good behaviour on its sites. In future work, analysis could be carried out to measure this benefit on a platform such as HeyStaks.

Research has indicated that publicly displaying information on reputation or trustworthiness of users helps to foster confidence in the site itself [71]. Research in recommendation systems has found that improving system transparency – giving users information about why the system made particular recommendations to them – may improve the system itself over time [41] and helps to foster trust in the system [12, 58]. If recommendations are informed by the reputation of their producers, it may be useful to make that information available to potential consumers. An interesting avenue

² http://www.giraph.apache.org
for future work is to examine how displaying reputation information to consumers might affect their decisions in selecting content that is recommended to them, and how providing this information changes users’ opinion of the system.

8.3.3 *Reputation and Robustness*

Malicious use is a well-documented problem in the area of recommender systems [43, 52]. Users can build a profile in order to skew recommendations that other users might receive, either to intentionally promote an item, or to decrease the likelihood of promotion. Various attack strategies have been identified, and modern recommender systems must be designed with these strategies in mind. Mobasher et al. [43] present a “semantically enhanced” recommendation algorithm, which uses the text provided about an item or piece of content to influence the recommendation process and thus mitigate the negative effects of malicious rating activity.

Reputation systems can be used to aid a website’s robustness, and research has shown the benefits a reputation system can bring in this regard [11, 26]. In recommender systems research, reputation has been used as a proxy for past recommendation success; specifically, if a user produces a piece of content that is subsequently selected on many subsequent occasions by other users, their reputation improves. Conversely, if users rarely select content they provide, their reputation suffers. This concept has been discussed by O’Donovan and Smyth [52], who feed this information back in to the recommender system to improve its accuracy at delivering recommendations a user might find useful. Subsequent work by the authors also reveal that incorporating reputation information into recommender systems can help to mitigate the negative effect malicious user activity has on the experience of others [53]. Specifically, including trust information can help minimise prediction shift if a malicious user is attempting to “push” a particular item by intentionally giving it a high rating while making their profile similar to others in an attempt to ensure the item gets recommended to them (known as an “average attack”).

However, reputation information must be used with caution, particularly when integrating it into a user interface. Early work that examines feedback-based reputation systems shows that interfaces which elicit information about transactions between two parties is often reciprocal, and so may be a poor indicator of performance in future transactions [59]. Another problem with transparent reputation systems is that the information they display may provide malicious users with information on how
to manipulate it. Robustness of reputation systems has been discussed in detail by Jøsang and Golbeck [25]. This work explores different techniques malicious users can employ to attack a reputation system, such as collusion, continuous re-entry and Sybil attacks (i.e. a user has multiple identities on the same site). The authors state that every reputation system is vulnerable to malicious use, thus motivating robustness analysis of our own model.

The above related work shows that reputation systems can be used to enhance a recommender system’s resistance to malicious use. However, for a proper evaluation, not only should the recommender systems’ robustness be tested, but the reputation system as well. Future work should then involve identification of strategies to attack the approach described in this thesis, and also how it might affect the robustness of any system which employs it.
Part IV

APPENDIX
APPENDIX

The work presented in this thesis is largely contained within published papers, and as such there were constraints on articulation of experimental methodologies and findings. This appendix presents information to supplement that presented in the papers, in the hope of aiding the reader’s understanding of the work.

A.1 PAPER 1 EXPERIMENT METHODOLOGY: IDENTIFICATION OF EXPERIMENTAL USERS

In this experiment, the reputation model proposed in the paper is applied using the activity performed by these users over the course of a nine-month period. This experiment was run at a time when HeyStaks was in private-beta, and as such, they were the only users that recorded a non-negligible degree of reputation when the model was applied to the larger HeyStaks community. Thus, while there were 26 users who accrued reputation, a total of 99 users participated in collaboration events as consumers and thus were providers of reputation, as per the model described.

![Figure 20: Network sizes for the 26 trial users in paper 1 (see Chapter 2).](image)

This is implied in Figure 20 (appearing as Figure 9(b) in the paper), which shows the biggest network size for a single experiment user was 89, indicating a larger
set of users in the HeyStaks community who provided the experiment users with reputation.

The paper states that the experiment users formed only a subset of the larger HeyStaks community, and users with negligible or no reputation were not considered in the experimental findings. This may indicate subjective bias in the results. However, the primary motivation of the paper is to explore the concept of reputation in the context of a real system, rather than to demonstrate the model’s efficacy, and as such considering only a subset of users does not change the results presented.

A.2 PAPERS 2–5 EXPERIMENT METHODOLOGY: GROUND-TRUTH VALIDATION

Paper 2, entitled “A Case-study of Collaboration and Reputation in Social Web Search”, describes a live-user trial, where users’ actions on search results are recorded as part of a quiz posed to them, and each result is manually checked for relevance to the quiz questions. Specifically, each of the 724 unique URL’s recorded during the experiment were individually checked against answers to quiz questions, and each URL was marked as not relevant, 1 partially relevant or relevant to the answers. This process was a manual one, carried out by four people furnished with answers to the quiz questions. This data is then used as the ground-truth in Papers 3, 4 and 5 in order to test the extent to which the reputation model positively influences the HeyStaks result recommendation process.

There are two key limitations to this approach which were not originally discussed in the paper. First, manual checking is always subject to human error. However, any potential problems arising from such an approach were mitigated by ensuring a consensus among the four checkers was reached for the relevance of each URL to the quiz questions. Secondly, for each URL recorded an assumption was made that the user(s) selecting it were intending on trying to answer a quiz question. This was considered during the process of choosing suitable quiz questions, specifically, questions were chosen based not only on their difficulty, but on their obscurity in the sense that simply typing the question in to Google would not return the answer. This meant that the URL’s containing the answers or partial answers were much more straightforward to identify.
A.3 PAPER 2: MISUSE OF THE TERM “SIGNIFICANT” TO DESCRIBE EXPERIMENT FINDINGS

Please note that in Paper 2, in Section 5.4 (page 21) and Section 5.7 (page 29), the term “significant” was misused. The results cited were not tested for any statistical significance. The term should be read as “notable” in these cases.


DECLARATION

I, Kevin McNally, declare that this thesis titled, ‘A Model of Collaboration-based Reputation for Social Recommender Systems’ and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.

- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.

- Where I have consulted the published work of others, this is always clearly attributed.

- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.

- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Dublin, Ireland, September 2013

Kevin McNally, September 3, 2014