The Experience Web: A Case-Based Reasoning Perspective

Barry Smyth[1], **Pierre-Antoine Champin**[1,2]
[1] CLARITY: Centre for Sensor Web Technologies
School of Computer Science and Informatics
University College Dublin
Belfield, Dublin 4, Ireland
barry.smyth@ucd.ie
[2] LIRIS, Université de Lyon, CNRS, UMR5205,
Université Claude Bernard Lyon 1, F-69622, Villeurbanne, France
pchampin@liris.cnrs.fr

Abstract

With the rise of user-generated content (blogs, wikis, ratings, reviews, opinions etc.) the web is evolving from a repository of content into a repository of experiences. And as it evolves there are many opportunities to harness these experiences. In this paper we consider some of the challenges associated with harnessing online experiences by adopting a case-based reasoning perspective, and highlighting how existing case-based approaches might be adapted to take advantage of this new world of the experience web.

Introduction

"Web 2.0", the "Social Web", the "Future Web", these are all terms and phrases that are commonly used to refer to the ever-changing world of the web. Today web content comes in many different forms, from simple HTML pages and Flash sites to sophisticated audio/video interactive media, and the tools that provide us with access are the ubiquitous search engines, which respond to billions of queries every day. In more recent times the web has evolved to accommodate a new type of content that is in some sense less explicit, more dynamic, and less permanent, than the traditional content of pages and sites. The arrival of blogs in 1999, as a simple way for users to express their views and opinions, ushered in this new era of *user-generated content* (UGC) as many sites quickly began to offer a whole host of UGC alternatives including the ability to leave comments, write reviews, as well as the ability rate or vote on the comments/opinions of others. The result has been an increased emphasis on people rather than content and, in combination with social networking services, this has precipitated the growth of the *social web* as a platform for communication and collaboration.

What has all of this got to do with *experiences*? The essential point is that the combination of traditional web content (the *digital artifacts* as the fundamental units of conventional web content) with the opinions, views, and ratings of users, is the very stuff of experiences (Plaza 2008); see Fig. 1. More generally, the combination of a digital artifact plus its *usage information* is an experience repository. Usage information encompasses not only the *explicit* forms of user-generated content mentioned above but also the *implicit* usage information, such as the navigation trails and/or search queries that led to a particular digital artifact, usage information that is recorded within server logs. Echoing the views of (Plaza 2008), the web is the ultimate decision-support tool and there is a significant opportunity to harness these many and varied types of experiential knowledge in order to help people make the right decisions using the right information at the right time.

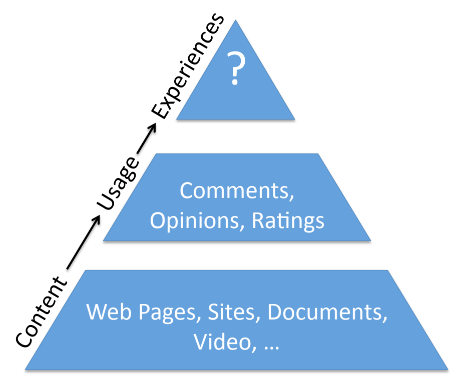

Figure 1: The Evolution of the Experience Web.

The ability to harness and reuse these online experiences has tremendous potential and the purpose of this paper is to explore how we might go about meeting this challenge, and to what end. Our starting point is the research of the *case-based reasoning* (CBR) community, where more than 20 years of research effort has been devoted to exploring different aspects of *reasoning from experiences* (Aamodt and Plaza 1994; Leake 1996; Watson 1998). The core CBR proposition is that new problems can be solved by reusing the solutions to similar problems that have been solved in the past. A typical CBR system will capture experiences in the form of individual *cases*. Each case will normally contain information about the particular problem it is designed to solve (the *specification* part of the case) and the details of the solution used to solve this problem (the *solution* part of the case). Sometimes, cases may also contain information about the *outcome* of the particular solution when it was applied. When faced with a new *target* problem the CBR system will *retrieve* one or more similar cases, and their solutions will be *adapted* to produce a new solution that fits the target problem. The resulting target specification-solution pair may be learned as a new case. In this way, CBR systems attempt to obviate the need for domain specific problems solving knowledge, replacing first-principles reasoning techniques with experience reuse.

The purpose of this paper is to identify challenges and pose questions rather than to propose answers through fully worked solutions. Specifically we wish to consider the type of tools and techniques that need to be developed in order to support (personal) experience reuse as a basic web service, in much the same way as web search is a basic service today. In particular we will focus on 3 core challenges as follows:

1. *Capturing Personal Experiences.* How might we capture, organise, and share the online experiences of web users? How can current tools and applications be augmented to accommodate experience capture and reuse, leading to the creation of shared personal case bases.

2. *Coping with Noise.* Personal experience creation is a departure from traditional approach to *expert-led* CBR (in which cases are created by domain experts or as a direct result of expert problem solving). Capturing the ad-hoc experience of individuals introduces a significant quality risk. Not only are our opinions and views subject to change, but the way they are collected on the web may not always reflect our own perception, introducing a considerable amount of noise. How might an experience reuse system cope with repositories of experiences that are extremely noisy?

3. *Reuse in Context.* How can we leverage the right experience at the right time and in the right context, bearing in mind that relevant experiences may be distributed across multiple cases and indeed case bases. In particular, understanding the provenance of a case and the reputation of the case creator — while dealing with the attendant privacy issues — will play a significant role in the development of *experience-based interfaces* that will integrate experience reuse into our everyday tasks.

In what follows we will make our discussion concrete by drawing on a particular implementation of one attempt to harness the experience web in the area of web search. HeyStaks is a *search utility* that is designed to work with mainstream search engines by allowing users to organise, share and reuse their particular search experiences; see www.heystaks.com for a live beta. It comes in the form of a browser toolbar and back-end server to provide users with an experience-based web search support system that is fully integrated with Google. While it will not be possible to describe the technical details behind HeyStaks the interested reader is referred to (Smyth et al. 2009) for further information.

Challenge 1 - Capturing Personal Experiences

Experience-like information is now commonplace on the web, as many sites and services attempt to supplement their core content with the opinions, ratings, and comments of users. The challenge for end-users is that these experiences

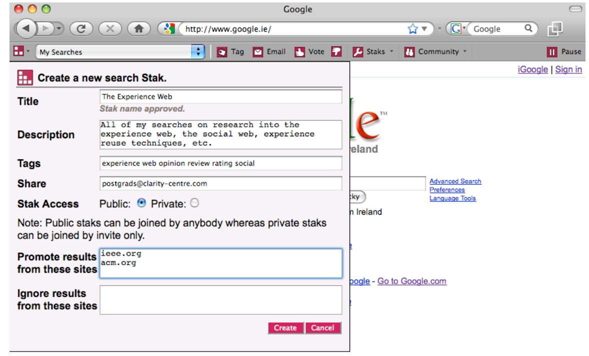

Figure 2: Creating a new search stak.

are often ad-hoc and usually fragmented. As a user my opinions and reviews are thinly spread across many different sites and the trusted opinions of my social network are all but invisible to me. How might we support individual users when it comes to the creation and sharing of their personal experiences? As a user, I want to be able to keep track of my experiences and the relevant experiences of my trusted friends and colleagues. I want to be reminded of similar experiences as I interact with services online; if I am booking conference accommodation (through the conference web site) I would like to be reminded if I have stayed at a particular hotel before, or if one of my colleagues has stayed there, especially if the experience was good or bad.

To meet this challenge there is the need for common representational formats as a way to represent digital artifacts; this has been a long-time goal of semantic web initiatives (Berners-Lee 1998). In addition, experience creation, organisation and sharing needs to be built into the very fabric of the web, and the tools that we use to interact with web services. In short, there is a need for experience creation and management tools that are as much part of the web experience as the browser and search engine are today. This is in contrast to the work of the case-based reasoning community which, to date, has focused on the the provision of dedicated CBR tools. These tools are mainly designed to be used by domain experts, allowing for the creation of standalone CBR systems and case bases. If we are to incorporate experience reuse into our online-lives then a different sort of approach is needed, one that sees experience management fully integrated into the many and varied tools and services that we naturally use, from search engines and portals to e-commerce services to online word processors etc (Mille 2006).

HeyStaks addresses these challenges in the domain of web search by allowing users to create repositories for search experiences related to a particular topic or task. Each repository is called a *search stak* and is effectively a case base of search cases. Each case corresponds to a single result page that has been 'selected' for this stak during a user's searches. Each case is anonymously associated with a number of implicit and explicit interest indicators, including: the total number of times the result has been selected during a search, the query terms that led to its selection, the snippet

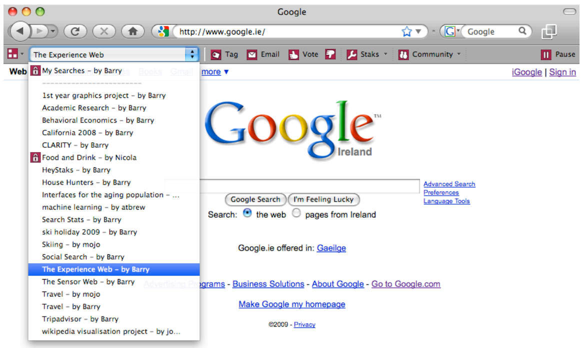

Figure 3: Selecting search stak prior to search.

terms associated with the result when it wasn't selected, the total number of times a result has been tagged and the terms used to tag it, the total votes it has received, and the number of people with whom it has been shared. In addition each term (query, tag, snippet) is linked to a hit-count that reflects the number of times that this term has been associated with the page in question.

For example, Fig. 2 shows how a user can use the HeyStaks toolbar to create a search stak to capture their searches related to the experience web. Then, as they search they can select a suitable stak prior to, or during a search, as a way to ensure that the current search experience is stored within the appropriate stak; see Fig. 3.

In this way users can create and share different repositories of search experiences. As they browse and search, these repositories are enriched with additional searches. For example, while browsing users can use the HeyStaks toolbar to vote on any particular page, or they can tag a page or share it with a friend. As they vote, tag, and share this information (tag terms, votes) is associated with the page in question in the current stak. In turn, as users search, their result click-thrus are taken as implicit indicators of interest so that click-thru frequency information is also associated with a given result for a given query in the current stak. Moreover, the HeyStaks toolbar augments conventional search result-lists to provide access to tagging, voting, and sharing actions at the level of individual results. For example, Fig. 4 shows the Google results for the query "stability clustering" in a "Machine Learning" stak created by a group of ML researchers. As the user mouses-over individual results popup HeyStaks icons provide access to voting, tagging, and sharing features as shown. The figure also shows how a number of results have been *promoted* by HeyStaks, which we will discuss in more detail in section .

The power of staks as search experience repositories comes to be fully felt when they are shared with others. This facilitates the aggregation of search experiences across groups of friends and colleagues. In the case of our Experience Web search stak, by sharing it with a group of interested searchers this stak will be added to their own toolbar, and will therefore quickly grow to accumulate a significant store of related search experiences as the basis for targeted promotions during future searches by stak members. In this

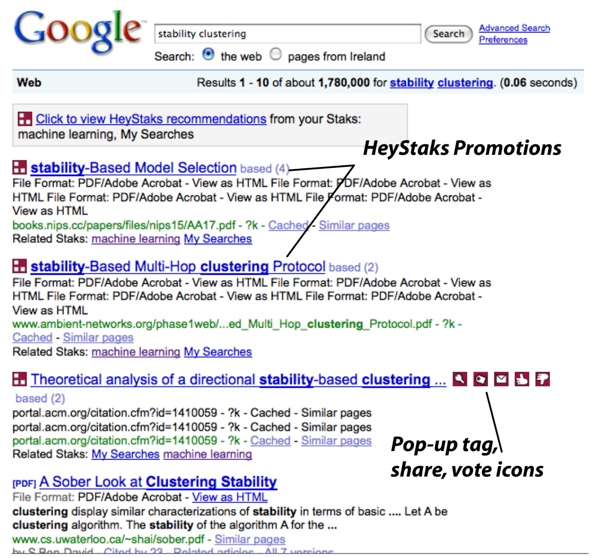

Figure 4: HeyStaks augments Google result-lists with a range of experience-capture functions and also highlights recommended results that are promoted because of the recent search experiences of friends and colleagues.

Figure 5: Summary statistics, related to stak creation and sharing; see (Smyth et al. 2009).

way, stak members will benefit from results found by other members for similar queries in the past.

In a recent beta deployment across 95 users, over a 3 month period, we found stak sharing to be commonplace. The average user created 3.2 search staks and joined a further 1.4 staks and 70% of users shared staks with at least some other users (see Fig. 5); see (Smyth et al. 2009) for further information.

It is worth highlighting at this point that the power of the HeyStaks browser toolbar is that it facilitates experience capture across a variety of services and search engines without the need for any service-level code-integration. Thus experiences act as a layer of knowledge that is independent of the underlying web content or services to which it relates, and provides for a very powerful and flexible service delivery platform.

Challenge 2 - Coping with Noise

In the previous section we argued for experience management and creation tools as a necessary feature of future web infrastructure, and we described HeyStaks as a point example of how this has been achieved in the context of web search. Providing for the capture of online experiences will ensure that our experience repositories grow quickly to reach some critical mass, especially if these experiences are created based on implicit as well as explicit actions and activities. For example, in the case of HeyStaks, every search by a user that results in at least one result click-thru is translated into an experience (search case). The problem now becomes one of quantity versus quality and, specifically, the extent to which these experiences will serve as a reliable basis for future actions and decision making. For example it is well known that our real life experiences are coloured by context and so our online ratings and opinions can be expected to vary significantly; we may find that, over time, the same user differently rates the same move, for instance.

Similarly it will not always be possible to infer the right context for a given experience so the resulting experience may be misrepresented or misclassified. This is a problem in HeyStaks, exacerbated by the need for users to select the current search stak at search time. If the user does not select the correct stak then their new search will be stored in whatever stak happens to be active at search time. This is in part addressed by using recommendation techniques to automatically select an appropriate stak (from the user's stak list) based on their current search query; briefly, the current query is matched against experiences in available staks and the stak with the best matching experiences is recommended. Thus at search time the user will be alerted to the fact that their current stak has been changed to one that better fits their query. Reliable recommendations cannot always be made, however, especially if there are few experiences in the user's staks as the basis for query matching, and so searches continue to be misclassified by HeyStaks.

Thus, a key challenge when it comes to personal experience capture and management concerns the ability to deal with potentially significant levels of noise. For a number of years the case-based reasoning community have looked a variety of techniques for editing experiences, under the heading of *case-base maintenance*; see for example the work of (Craw, Massie, and Wiratunga 2007; Leake et al. 2001; Leake and Wilson 1998; Smyth and McKenna 2001b). However, existing techniques are usually designed to manage case bases with relatively low amounts of noise and work best with cases where there is an objective measure of when a case can be used to correctly *solve* some target problem. These assumptions are less likely to hold in the experience web. For example, in HeyStaks we currently find a significant number of search experiences to be stored in an incorrect stak; largely due to failings in the stak recommendation feature mentioned above and because many users "forget" to select an appropriate stak at search time.

One technique for coping with high-levels of experience noise is to identify what we call the *experience kernel*. In the case of HeyStaks this is a subset of documents (cases) that are assumed to be relevant to that stak; related to the notion of a *competence footprint* in the work of (Smyth and Keane 1995; Smyth and McKenna 2001a). Experience kernels can be calculated using a variety of techniques. Because of the extent of noise within search staks we have explored a number of different approaches to based on the clustering of documents (cases) into meaningful groups of related experiences. For the purpose of this paper we will briefly review the simplest *term-based clustering* approach here. Two query terms used in a stak are considered related if at least n documents contain both of these terms[1]. Briefly, we use a complete linkage clustering algorithm (Romesburg 2004) to build clusters of related terms according to this measure. Then, we remove all clusters containing less than s terms. Both thresholds n and s are adapted for each stak in order to obtain a non empty set of typical terms. The kernel of the stak is composed of all the documents in that stak that were retrieved with at least one typical term.

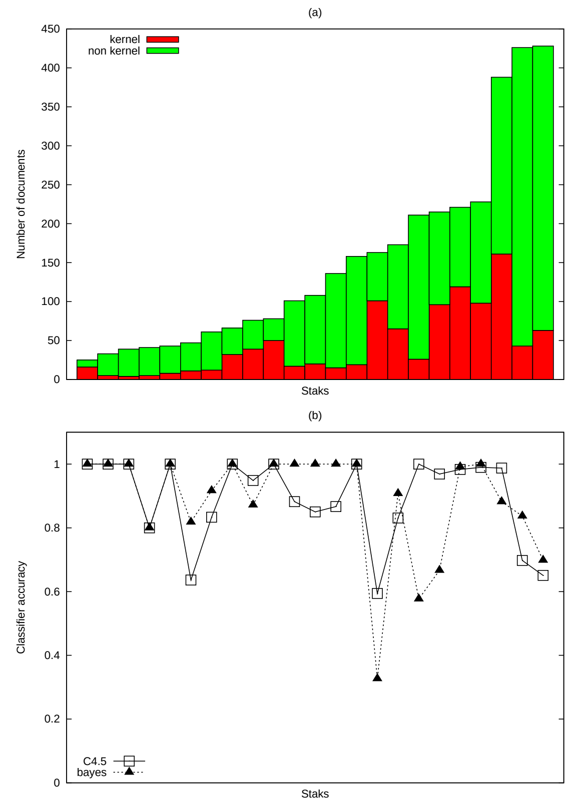

Figure 6: (a) number of kernel and non-kernel documents and (b) accuracy of the classifiers, for each test stak.

An interesting feature of this method is that, although it retains a small number of terms (at most 15% for some staks, and 5% globally), it keeps a reasonably high number of documents in the kernel. For example, Fig. 6(a) shows the relative number of documents contained in a subset of staks, including the size of each stak's kernel. The relative size of the kernels gives a clear indication of the amount of (potential) noise that is contained within staks with some staks being dominated by potential noise while others enjoy much larger kernels.

The ability to identify reliable experience kernels leads to a number of ways to improve the manner in which experiences are captured and organised. For example, we can

[1]More precisely, n is weighted by how many times each term was used to retrieve each document.

construct a classifier from the experience kernels and use this classifier to predict the right stak for a given search experience. In this scenario each instance corresponds to a document from the relevant stak kernel with the stak id used as the class. For instance, from the staks used in Fig. 6(a) we constructed a C4.5 (Quinlan 1993) decision tree and a bayesian network, from the kernel documents, to use as test classifiers. Using a standard 10-fold cross validation evaluation delivers stak-by-stak classification results shown in Fig. 6(b); the average accuracy is 89% for the decision tree and 83% for the bayesian network.

The predictive power of both classifiers is generally good. Both mostly agree on the "difficulty" of certain staks – which is not trivially correlated with other parameters, such as the kernel size or the overlap of typical terms with other staks. This suggests that the kernel building technique is capable of identifying collections of core documents that are at least reasonably predictable within a stak, and lends confidence to the prospect of using the classification approach as a way to associate non-kernel documents with their "correct" staks as part of a maintenance process.

These preliminary results merely scratch the surface of some of the maintenance challenges associated with the experience web. They serve to highlight the potential for high degrees of noise in personal experience repositories where the inadvertent actions of the user can lead to experiences being misclassified. They also point in the direction of a potential solution since if experience kernels can be reliably identified then they can also be used to guide experience maintenance. In the future it may be interesting to quantify how mature or consensual a stak has become, or on the contrary to detect when it is subjected to an abrubt change. This kind of information could provide additional context when using those staks for recommendation.

Challenge 3 - Reuse in Context

In this paper we have argued that experiential information is a natural by-product of our online activities and these experiences can be captured and stored with the potential for future reuse as a way to support and improve our decision making. Our web search case-study presents one concrete example of how, during the natural course of our web searches we are capturing and sharing actual search experiences. In this section we consider some of the reuse challenges that must be addressed when it comes to actually putting stored experiences to good use. Once again there is an integration challenge, related to how relevant experiences might be incorporated into a particular application interface. In addition, there is the obvious challenge of experience selection (and ranking) as the right experiences need to be chosen at the right time and in the right context. Moreover, we note that when experiences are created and shared within communities of users, experience reuse creates a new form of collaboration network between community members.

When it comes to the integration issue, the essential challenge is about how best to incorporate experience in order to guide our online tasks. In web search there are a variety of ways in which our past experiences, and those of others, can be used to guide searching. Already, for example, search engines like Google harness simple forms of experience to make query suggestions[2], by recommending additional terms that have commonly co-occured with a given set of target query terms.

More directly, however, experiences can be reused as a way to recommend actual result-pages during web search, and this is the approach adopted by HeyStaks: individual pages that have been frequently selected, tagged or voted on, for similar queries are highlighted, promoted, or inserted directly into the Google result-list by the HeyStaks toolbar. Briefly, to generate these promotion candidates, HeyStaks uses the current query as a probe into each stak, to identify a set of relevant cases. Each candidate case is scored using a similar technique to that described by (Boydell and Smyth 2007) by using a TFIDF (*term frequency • inverse document frequency*) function as the basis for an initial recommendation ranking; this approach prefers cases that match terms in the query which have occurred frequently in the case, but infrequently across the case base as a whole; see also (Smyth et al. 2009) for a more detailed analysis of HeyStaks' promotion mechanism. A typical result is shown in Fig. 4 where we see a number of HeyStaks promotions that have been highlighted within the standard Google result-list.

In addition, however, HeyStaks is designed to explore another form of reuse, at the level of the case base, rather than the individual case. In this context, for a given target query, and in addition to the results that may be promoted from the currently active stak, HeyStaks will also consider experiences that are stored in other staks that the user is a member of, with a view to identifying relevant experiences in these alternative contexts. Consequently, HeyStaks can also recommend to the user a list of alternative staks as a source of further recommendations. Indeed, HeyStaks can also recommend public staks from the wide HeyStaks community — staks that the user has not yet joined — if they also contain similar experiences to the current search context.

Many of the recommendations that are made to the current user may come from their own personal search histories but many also come from the search histories of other users who also participate in their shared staks. This type of collaboration is commonplace within HeyStaks as the results of a recent beta deployment demonstrate. For example, a *net producer* is defined as a user who has *helped* more other users than they themselves have been helped by; in other words they have contributed more search experiences, which have been reused by others, than they themselves have reused. Conversely, a *net consumer* is defined as a user who has been helped by more users than they themselves have helped; in other words they tend to benefit a lot from the experiences of others but dont contribute many new experiences of their own for others to benefit from. Fig. 7 shows that 47% of HeyStaks users are net producers. In other words almost half the users are helping others, by their search experiences, more often than they themselves are helped in return: social search is inherently altruistic. Indeed, when we look at the promotions that users actually select during their searches

[2] `https://adwords.google.com/select/KeywordToolExternal`

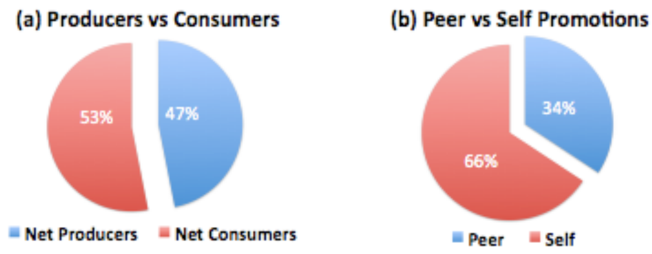

Figure 7: Summary statistics, related to search collaboration; see (Smyth et al. 2009).

we find that, on average, 33% of these are so-called *peer promotions*, promotions that are directly derived from the experiences of others, where as 66% are so-called *self promotions*, promotions that come from the searcher's own personal experiences. In this way, experience reuse in HeyStaks pervades conventional web search as results and staks (case bases) are suggested, on the fly, to searchers.

Conclusions

The web provides a rich source of explicit and implicit experiences but, by and large, these experiences are either diluted across a great many different sites and services, or never captured in the first place. From a case-based reasoning perspective we know how to represent and reuse experiences and so there is considerable opportunity for the CBR community to turn its attention to the web as a new source of experiential knowledge that is just waiting to be harnessed. In this paper we have taken the first tentative steps in this regard, in an attempt to explore this type of experience reuse, and the challenges that it presents.

Our vision is one that reflects a bottom-up approach to experience reuse. We have argued the need for experience capture and reuse facilities to be integrated into our online tools and services, so that individual users can benefit from their own past experiences to begin with. We have also argued the need for experiences to be shared among groups of related users and interested parties, so that people can benefit from aggregate community experiences. We have also highlighted issues of reliability and noise when it comes to personal experience capture and argued the need for new techniques to cope with high degrees of experience noise that would be considered to be unusual in a conventional expert-created case base.

Throughout the paper we have attempted to provide concrete examples with reference to one particular online experience reuse system that has been deployed in the domain of Web search. As such, the HeyStaks system illustrates many of the points that have been made. It integrates experience capture and reuse as part of the traditional web search interface and allows for the creation and sharing of personal search experiences. These experiences are inherently noisy and we have described one approach to coping with this noise by identifying experience kernels within a case base. Finally we have demonstrated how current HeyStaks users are benefiting from their own search experiences and those of others, leading to an effective form of search collaboration as a side-effect of experience reuse and sharing.

Acknowledgements

Based on works supported by Science Foundation Ireland, Grant No. 07/CE/I1147, the French National Center for Scientific Research (CNRS), and HeyStaks Technologies Ltd.

References

Aamodt, A., and Plaza, E. 1994. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Commun.* 7(1):39–59.

Berners-Lee, T. 1998. Semantic web roadmap. Draft, W3C.

Boydell, O., and Smyth, B. 2007. Enhancing case-based, collaborative web search. In *ICCBR*, 329–343.

Craw, S.; Massie, S.; and Wiratunga, N. 2007. Informed case base maintenance: A complexity profiling approach. In *AAAI*, 1618–1621.

Leake, D. B., and Wilson, D. C. 1998. Categorizing case-base maintenance: Dimensions and directions. In *EWCBR*, 196–207.

Leake, D. B.; Smyth, B.; Wilson, D. C.; and Yang, Q. 2001. Special issue on case-based maintenance. Computational Intelligence.

Leake, David, B. 1996. *Case-Based Reasoning: Experiences, Lessons and Future Directions*. Cambridge, MA, USA: MIT Press.

Mille, A. 2006. From case-based reasoning to traces-based reasoning. *Annual Reviews in Control* 30(2):223232. Journal of IFAC.

Plaza, E. 2008. Semantics and experience in the future web. In *ECCBR*, 44–58.

Quinlan, J. R. 1993. *C4. 5: programs for machine learning*. Morgan Kaufmann.

Romesburg, C. 2004. *Cluster analysis for researchers*. Lulu. com.

Smyth, B., and Keane, M. T. 1995. Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. In *IJCAI*, 377–383.

Smyth, B., and McKenna, E. 2001a. Competence guided incremental footprint-based retrieval. *Knowl.-Based Syst.* 14(3-4):155–161.

Smyth, B., and McKenna, E. 2001b. Competence models and the maintenance problem. *Computational Intelligence* 17(2):235–249.

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

Watson, I. 1998. *Applying case-based reasoning: techniques for enterprise systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.