RerankEverything: A Reranking Interface for Browsing Search Results

Takehiro Yamamoto Satoshi Nakamura Katsumi Tanaka

Department of Social Informatics, Graduate School of Informatics, Kyoto University
Yoshida-Honmachi, Sakyo, Kyoto 606-8501, Japan
+81-75-753-5969
{tyamamot, nakamura, tanaka}@dl.kuis.kyoto-u.ac.jp

ABSTRACT
This paper proposes a system called *RerankEverything*, which enables users to rerank search results in any search service, such as a Web search engine, an e-commerce site, a hotel reservation site and so on. In conventional search services, interactions between users and services are quite limited and complicated. In addition, search functions and interactions to refine search results differ depending on the services. By using *RerankEverything*, users can interactively explore search results in accordance with their interests by reranking search results from various viewpoints.

Categories and Subject Descriptors
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *Graphical user interfaces, Interaction styles.*

General Terms
Design, Human Factors.

Keywords
Search user interfaces, reranking, wrapper generation.

1. INTRODUCTION
Nowadays, people use many kinds of search services such as Web search engines, e-commerce sites, hotel reservation sites, video sharing sites and so on. These search services have become important information resources for many people. When using a search service, users usually issue a query to it. The service then selects contents from its index, ranks them according to its own criteria and finally displays them as search results.

However, it is not always easy for users to obtain satisfactory information from search results. First problem is that it is difficult for users to create appropriate queries that clearly describe their information needs [2]. Since users' information needs are often not clearly defined, their queries are often too short and ambiguous. Therefore, it is difficult for search services to estimate users' information needs only from received queries and return good rankings that will satisfy all users. Second problem is that search functions and interactions to refine search results differ depending on the services. These fundamental problems limit users to browsing search results passively. If users were able to flexibly refine search results based on their own criteria, they could obtain more useful content in accordance with their interests. Our goal is to realize such an interactive search system.

In this paper, we propose a system called *RerankEverything*, which enables users to rerank search results in any search service. In the system, users can intuitively rerank results by simply selecting an interesting term, or a numeric value of a search result. Our system provides users not only the reranking interface, but also a tag cloud to encourage users to explore search results from various viewpoints, and a simple interface to specify an html element that contains a search result to recognize structures of the search results page.

WWW 2010, April 26-30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

2. SYSTEM ARCHITECTURE
We implemented our system (*http://rerank.jp/rerankeverything.xpi*) as an extension to the Mozilla Firefox Web browser. Our system mainly consists of three components. The overview of our system is showed in Figure 1.

2.1 Reranking Interface
In our system, users can give both term-based feedback and numerical feedback by using the same interaction style, simply selecting an interesting part of a search result. When a user selects a certain term and presses the upgrade/downgrade button, the system then reranks the results by upgrading/downgrading results that contain the selected term. Similarly, a user can sort search results according to a selected numerical attribute. Figure 2 illustrates how the user reranks search results in the publication search result according to the number of citing counts.

2.2 TermCloud
To encourage users to explore search results from various viewpoints, our system displays *TermCloud* [3], which are generated from terms that frequently appear in search results. Users can give term-based feedback and rerank search results by simply clicking a term in a *TermCloud*. Figure 3 shows an example of a *TermCloud* of Web search results for the query "pork green pepper recipes." In that case, the system displays some terms that are related to cooking like "garlic", "onion", "Chinese", "easy" and so on. Because a *TermCloud* might bring interesting and various terms in search results to the attention of users, we expect users will be more likely to find valuable and diverse search results.

2.3 Wrapper Generation Interface
The number of existing search services is huge and the structures of search results vary according to the services. To detect a structure of search results in an HTML source of a target page, we need to prepare wrappers for each service. Our system provides an interface to generate wrappers based on simple user interactions. The interface also works interactively inside a Web browser and requires no special knowledge of DOM, JavaScript, XPath, etc.

Our idea is to visualize nodes that the system currently recognizes as search results while a user moves the mouse cursor. Figure 4 illustrates how a user would generate a wrapper that works on the Google Web search engine. As the user moves the mouse cursor, the system displays visual feedback by forms of rectangle indicating which nodes the system recognizes as search results.

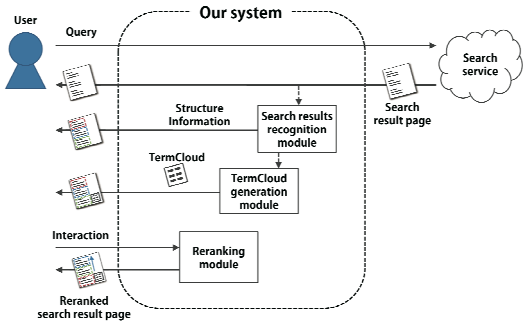

Figure 1. Overview of our system.

Figure 2. Reranking publication results by citing counts.

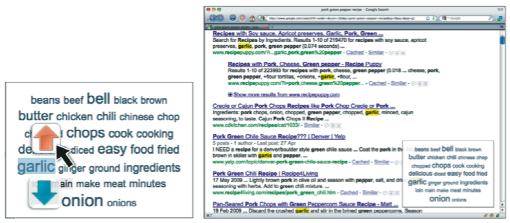

Figure 3. Example *TermCloud* for Web search results of query "pork green pepper recipes".

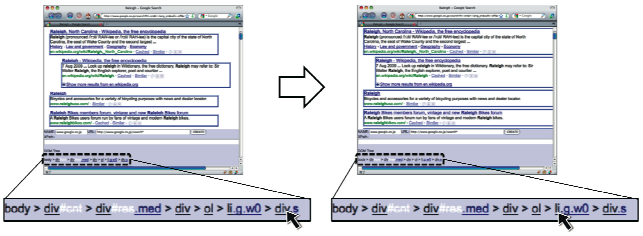

Figure 4. Wrapper generation interface.

3. EXPERIMENTS

First, we evaluated the execution time of our system. We measured the execution time for each step as shown in Table 1. In the experiments, we used Yahoo! Web search, setting the number of search results in a page to 10, 40 or 100. We measured the execution times for each step ten times. The preparation represents the steps from detecting search results in a target page using XPath to finishing to bind mouse events. For the reranking using term-based feedback, we randomly selected a term from a *TermCloud* for each reranking. For the reranking using numerical feedback, we used the last modified date to sort search results

Table 1 shows the average execution times for each step. The system was able to complete the preparation and reranking within 200 ms each, even when the number of search results was 100. Notably, when reranking using term-based feedback, the system could return reranked search results within much less than 100 ms. In other words, the execution times of upgrading or downgrading search results are much less than those of issuing new queries, which would require additional Web access. The execution time for generating a *TermCloud* is a bit longer than for the other steps. However, users can browse and rerank search results without waiting for the generation of the *TermCloud*.

Table 1. Execution time analysis.

	# of search results in a page		
	10	**40**	**100**
Preparation	23.3 ms	76.5 ms	180.6 ms
TermCloud generation	232.8 ms	299.7 ms	414.8 ms
Reranking (term-based)	13.6 ms	40.7 ms	54.1 ms
Reranking (numerical)	26.7 ms	91.8 ms	189.8 ms

Next, we investigated how many search services our wrapper generation method would work with correctly. In the experiments, we tested for the 20 search services listed in the Japanese Wikipedia article, *Search Engine*. This list included Google, Yahoo!, Baidu, Bing and so on.

Our method could correctly detect search results for 15 of the 20 services when only mouse operations were used. For three of the remaining five services, our method worked partially when only mouse operations were used. For two of these services, our system detected not only target search results, but also another contents such as advertisements. In these cases, our system could detect search results correctly when we manually changed the XPath by using the keyboard. The third of these services displays its search results using a two-column layout. Therefore, our system detected only half of the search results in the page.

For the last two services, our system failed to detect search results because the search results consisted of sibling nodes and a search result was not wrapped by a tag. Our system cannot detect such a structure even if we manually change the XPath by using the keyboard since our wrapper generation system can handle only one XPath. To handle these search services, we plan to modify our system and interfaces so that users can specify multiple nodes that represent a search result unit.

4. CONCLUSION

We have developed a system called *RerankEverything*, which provides unified and simple interaction techniques to enable users to rerank search results. In the future, we plan to distribute our system widely, collect reranking logs from many users and analyze these logs to investigate how our system would affect users' search behaviors. We believe our system has potential to bring users more diverse and serendipitous information [1].

5. ACKNOWLEDGEMENTS

This work was supported in part by Grant-in-Aid for JSPS Fellows (#09J55302), "Informatics Education and Research Center for Knowledge-Circulating Society" (Project Leader: Katsumi Tanaka, MEXT Global COE Program, Kyoto University), and by MEXT Grant-in-Aid for Scientific Research on Priority Areas: "Cyber Infrastructure for the Information explosion Era", "Contents Fusion and Seamless Search for Information explosion" (Project Leader: Katsumi Tanaka, A01-00-02, Grant#: 18049041).

6. REFERENCES

[1] R. W. White, B. Kules and S.N. Drucker. Supporting exploratory search. *Communications of the ACM*, 49(4), pp. 36–39, 2006.

[2] R. W. White and D. Morris. Investigating the querying and browsing behavior of advanced search engine users. In *SIGIR'07*, pp. 255–262, 2007.

[3] T. Yamamoto, S. Nakamura, and K. Tanaka. TermCloud for enhancing Web search. In *WISE'09*, pp. 159–166, 2009.