Applications of Web Query Mining

Ricardo Baeza-Yates

Center for Web Research, Dept. of Computer Science,
Universidad de Chile, Santiago
ICREA Research Professor, Technology Department,
Universitat Pompeu Fabra, Spain
rbaeza@dcc.uchile.cl, ricardo.baeza@upf.edu

Abstract. Server logs of search engines store traces of queries submitted by users, which include queries themselves along with Web pages selected in their answers. The same is true in Web site logs where queries and later actions are recorded from search engine referrers or from an internal search box. In this paper we present two applications based in analyzing and clustering queries. The first one suggest changes to improve the text and structure of a Web site and the second does relevance ranking boosting and query recommendation in search engines.

1 Introduction

Nowadays, search tools are crucial for finding information on the Web. They range from Web search engines such as Google and Yahoo! to search boxes inside a Web site. However, the amount of information available to us in the Web is continuously changing and growing, and thus search technology is continuously being pushed to the limit. Several new techniques have emerged to improve the search process, and one of them is based on the analysis of query logs. Query logs register the history of queries submitted to the search tool, and the pages selected after a search, among other data.

The main goal of this paper is to show how valuable is to perform log query mining, by presenting several different applications of this idea combined with standard usage mining. Although past research in query mining has focused in improving technical aspects of search engines, analyzing queries has a broader impact in Web search and design in two different aspects: *Web findability* and *information scent*. Web findability or Web ubiquity is a measure of how easy to find a Web site is, where search engines are the main access tools. To improve findability there are several techniques, and one of them is to use query log analysis of Web site search to include on the Web site text the most used query words. Information scent [24] is how good it is a word with respect of words with the same semantics. For example, polysemic words (words with multiple meanings) may have less information scent. The most common queries are usually the ones with more information scent, so analyzing Web search queries we can find words that are found in a site but have more or similar information scent than words in the home page (which have to be replaced or added); and words that are not found that imply new information to be included [2].

The search for certain groups of queries capturing common sets of preferences and information needs has been a recent trend in query log analysis [11, 36, 42]. Groups of related queries can be discovered by clustering queries using their related data in query logs. The clusters can then be used to improve search engines in several aspects. For example, search engines such as Lycos, Altavista, and AskJeeves, recommend related queries to the query submitted by a user. The related queries are computed by running query clustering processes. However, there is not much public information on the methods they use to do so.

A central problem that arises is how to represent the information needs represented by a query. Queries themselves, as lists of keywords, are not always good descriptors of the information needs of users. One reason for this is the ambiguity carried by polysemic words. On the other hand, users typically submit very short queries to the search engine, and short queries are more likely to be ambiguous. In order to formulate effective queries, users may need to be familiar with specific terminology in a knowledge domain. This is not always the case: users may have little knowledge about the information they are searching, and worst, they could be not even certain about what to search for.

The definition of an adequate way to model semantics of queries and similarity for queries is still an open problem. Query logs can heavily help in doing so. Previous works have proposed models to represent information needs related to queries based on data in query traces. We use this term to refer to the successive submissions of a query (usually by different users) in a period of time, along with the sets of URL's selected for them.

Another inherent problem to the use of query logs is that the clicked URL's are biased to the ranking algorithm of the search engine. The number of preferences on a particular answer depends on the position of the page in the answer list. Each successive page in the list is less likely to be selected by users. An adequate modeling for the preferences of users and for the semantics of queries should incorporate a method for reducing the bias caused by the current ranking produced by the search tool.

Our first example is a model for mining web site queries, usage, content and structure [2, 9]. The aim of this model is to discover valuable information for improving web site content and structure, allowing the web site to become more intuitive and adequate for the needs of its users. This model proposes the analysis of the different types of queries registered in the usage logs of a web site, such as queries submitted by users to the site's internal search engine and queries on external search engines that lead to documents in the web site. The words used in these queries provide useful information about topics that interest users visiting the web site and the navigation patterns associated to these queries indicate whether or not the documents in the site satisfied the queries submitted by the users. This model also provides a way to visualize and validate the web site's organization given by the links between documents and its content, as well as the correlation of the content to the URLs selected after queries.

Our second example is a new framework for clustering query traces [3, 4]. The clustering process is based on a term-weight vector representation of queries, obtained by aggregating the term-weight vectors of the selected URL's for the query. The construction of the vectors includes a technique to reduce and adjust the bias of the preferences to URL's in query traces. The vectorial representation leads to a a similarity measure in which the degree of similarity of two queries is given by the fraction of common terms in the URL's clicked in the answers of the queries. This notion allows to capture semantics connection between queries having different query terms.

Because the vectorial representation of a query trace is obtained by aggregating term-weight vectors of documents, this framework avoids the problems of comparing and clustering sparse collection of vectors, a problem that appears in previous work. Further, our vectorial representation of query traces can be clustered and manipulated similarly to traditional document vectors, thus allowing a fully symmetric treatment of queries and documents. This is important since we need to compute similarities between queries and documents for two cases: query recommendation and ranking boosting. The use of query clustering for query recommendation has been suggested by Beeferman and Berger [11], however as far as we know there is no formal study of the problem. Regarding ranking boosting, we are not aware of formal research on the application of query clustering to this problem. For both applications we provide a criterion to rank the suggested URL's and queries that combines the *similarity* of the query (resp. URL) to the input query and the *support* of the query (resp., URL) in the cluster. The support measures how much the recommended query (resp. URL) has attracted the attention of users. The rank estimate the *interest* of the query (resp., URL) to the user that submitted the input query. It is important to include a measure of *support* for the recommended queries (resp., URL's), because queries (URL's) that are useful to many users are worth to be recommended in our context.

The remainder of this paper is organized as follows. Section 2 presents the state of the art on query mining. In Section 3 we present a tool based on queries to improve a Web site. Section 4 shows the query clustering framework and its application to search engines. Finally, in Section 5 we outline some prospects for future work.

2 State of the Art

2.1 Characterizing Queries and User Behavior

Queries, as words in a text, follow a biased distribution. In fact, the frequency of query words follow a Zipf's law with parameter α, that is, the i-th most frequent query has $O(i^{-\alpha})$ occurrences. The value of α ranges from 0.6 [26] to 1.4 [7], perhaps due to language and cultural differences. However, this is less biased than Web text, where α is closer to 2. The standard correlation among the frequency of a word in the Web pages and in the queries is 0.15, very low [7]. That is, words in the content of Web pages follow a Zipf's distribution which order is very different from the distribution of query words. This implies that what people search is different from what people publish in the Web.

Table 1. Query statistics for four search engines

Measure	AltaVista	Excite	Fast	TodoCL
Words per query	2.4	2.6	2.3	1.1
Queries per user	2.0	2.3	2.9	–
Answer pages per query	1.3	1.7	2.2	1.2
Boolean queries	<40%	10%	–	16%

The search engine log also registers the number of answer pages seen and the pages selected after a search. Many people refines the query adding and removing words, but most of them see very few answer pages. Table 1 shows the comparison of four different search engines [29, 37, 33, 34, 35]. Clearly, the default query operation is dominant (in the case of TodoCL only 15% are phrase queries).

In addition, as an empirical study shows [29], the average number of pages clicked per answer is very low (around 2 clicks per query). Our own data shows the same. From navigational studies inside TodoCL we have found that advanced search is not used (but we must have it!), and less than 10% of the users browse the directory[1]. This means that instead of posing a better query, a trial and error method is used. Further studies of queries have been done for Excite [32, 33] and Fast [34], showing that the focus of the queries has shifted the past years from leisure to e-commerce. Other papers are focused in user behavior while searching, for example detecting the differences among new and expert users or correlating user clicks with Web structure [17, 6, 25].

2.2 Usage Mining for Web Site Design

Web site design based in usage mining is also called *user-driven design*. Previous work using Web mining for improving web sites, include the analysis of frequent navigational patterns and association rules based on the pages visited by users. In [10] they studied this in an on-line newspaper, with the main goal of discovering related newspaper sections, from the users points of view. In [18] new approaches are discussed for modeling user sessions and cluster analysis obtained from access logs.

WebSIFT [12] is a Web mining tool created to find interesting rules and patterns in a web site, defining as "interesting" rules and patterns that are new and unexpected. Other tools dedicated to the improvement of web sites using Web mining techniques are [21, 30, 31], most of these focused on improving the navigation (and sometimes the structure) of a web site dynamically and individually for each visitor. Also, [23] presents a method for mining patterns efficiently from access logs, and [40] improves the performance of the internal search engine.

Queries submitted to search engines can be a valuable tool for improving a web site. In [13] a method is proposed for analyzing similar queries on search

[1] TodoCL uses ODP (dmoz.org) as Google.

engines. The idea is to find queries that are similar to ones that directed traffic to a web site. This queries can contribute as new words for describing documents in the web site. Another kind of analysis, is the one presented in [2], that consists of studying queries submitted to a site's internal search engine. This approach proposes that valuable information can be discovered by analyzing the outcome of each query, in other words, if the user followed any of the results displayed by the search engine or not, or there was no answer.

2.3 Query Mining in Search Engines

Generic usage mining in search engines is surveyed in [8]. Few papers deal with the use of query logs to improve search engines, because this information is usually not disclosed. The exceptions deal with strategies for caching the index and/or the answers [38, 26, 20, 7] and query clustering as we see next. Recently, [27] weight different words in the query to improve ranking.

Some ranking algorithms such as DirectHit, PageRate [14], and MASEL [42] have included the analysis of query logs. Click-through data is also used by search engines to evaluate the quality of changes to the ranking algorithm by tracking them. DirectHit uses previous session logs of a given query to compute ranks based on the popularity (number of clicks) of each URL that appears in the answer of the query. This approach only works for queries that are frequently formulated by users, because less common queries do not have enough clicks to allow significant ranking scores to be calculated. For less common queries, the DirectHit rating provides a small benefit. Zhang and Dong [42] propose the MASEL (Matrix Analysis on Search Engine Log) algorithm which uses search engine logs to improve image search ranking. Clicks are considered positive recommendations on pages. The basic idea is to extract from the logs, relationships between users, queries, and pages. These relationships are used to estimate the quality of answers, based on the quality of related users and queries. The approach relies in the identification of users of a search engine, a task difficult to achieve in practice.

There is also recent related work on query clustering, and some approaches also consider data in query traces. Wen *et al* [36] propose to cluster similar queries to recommend URLs to frequently asked queries of a search engine. They use four notions of query distance: (1) based on keywords or phrases of the query; (2) based on string matching of keywords; (3) based on common clicked URLs; and (4) based on the distance of the clicked documents in some pre-defined hierarchy. Befferman and Berger [11] also propose a query clustering technique based on distance notion (3). This approach has limitations when it comes to identifying similar queries, because two related queries may output different URL's in the first places of their answers, thus inducing clicks in different URL's. As queries are short and the number of clicks is low, notions (1)-(3) are difficult to deal with in practice, because distance matrices between queries generated by them are very sparse. Notion (4) needs a concept taxonomy and requires the clicked documents to be classified into the taxonomy as well.

Fonseca *et al* [16] present a method to discover related queries based on association rules. Here queries represent items in traditional association rules.

The query log is viewed as a set of transactions, where each transaction represent a *session* in which a single user submits a sequence of related queries in a time interval. The method shows good results, but two problems arise. First, it is difficult to determine sessions of successive queries that belong to the same search process; on the other hand, the most interesting related queries, those submitted by different users, cannot be discovered. This is because the support of a rule increases only if its queries appear in the same query session, and thus they must be submitted by the same user. Zaiane and Strilets [41] present a method to recommend queries based on seven different notions of query similarity. Three of them are mild variations of notion (1) and (3). The remainder notions consider the content and title of the URL's in the result of a query. Their approach is intended for a meta-search engine and thus none of their similarity measures consider user preferences in form of clicks stored in query logs. Another approach adopted by search engines to suggest related queries is *query expansion* [5, 39]. The basic idea here is to reformulate the query such that it gets closer to the term-weight vector space of the documents the user is looking for.

3 Improving Web Site Design by Mining Queries

The Web has been characterized by its rapid growth, massive usage and its ability to facilitate business transactions. This has created an increasing interest for improving and optimizing web sites to better fit the needs of its visitors. Its more important than ever for a web site to be found easily in the Web and for visitors to reach effortlessly the contents they are looking for. Failing to meet these goals can result in the loss of many potential clients. In fact, this is an iterative process that can be modeled as in figure 1. Real usability and usage mining can only happen if the site is ubiquitous, that is, can be found easily by search engines first, and by people later.

Web servers register important data about the usage of a web site, this information generally includes visitors navigational behavior, the queries made to the web site's internal search engine (if one is available) and also the queries on external search engines that resulted in requests of documents from the web site.

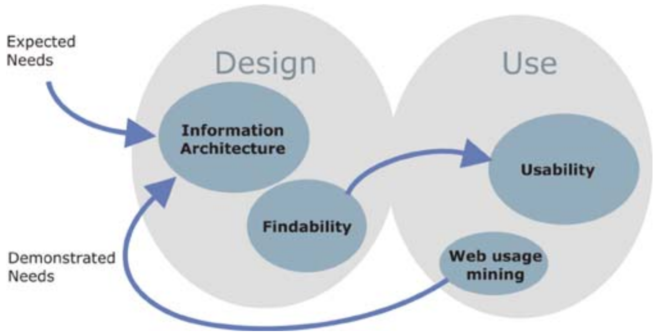

Fig. 1. Causal iterative model for Web design

All this information is provided by visitors implicitly and can hold the key to significantly optimize and enhance a web site.

This section presents a model for mining usage, content, and structure within a web site, centered in queries, to discover new and interesting information regarding ways to improve it [2, 9]. The model also allows to carry out a validation of the site's content organization in relation to the link organization between documents, as well as the URLs selected due to queries. The output consist of several reports and visualizations that make possible for the site's webmaster to decide how to modify the web site, which we do not cover here.

The suggestions generated by the model consist of: adding new contents to the site or broadening the current coverage of certain topics, changing or adding words to the hyperlink descriptions, adding new links between related documents, revise links between unrelated documents, and check the consistency of the content and the URLs selected after queries.

This model works on all types of web sites but is specially useful on large sites, in which the content is hard to manage for the site administrator, by pointing out the possible "problem areas" and ways to solve these conflicts, and improve its organization.

3.1 Model Description

This model develops different usage, structure and content mining tasks. Its input is the web site's access logs and the structure and content of its pages. The structure of the web site is obtained from the links between the documents and the content corresponds to the text associated to each document.

Figure 2 shows the description of the model, which gathers information about internal and external queries, navigational patterns and links in the web site to discover information scent that can be used to improve the site's contents. Also the link and content data from the web site is analyzed using clustering of similar documents and connected components.

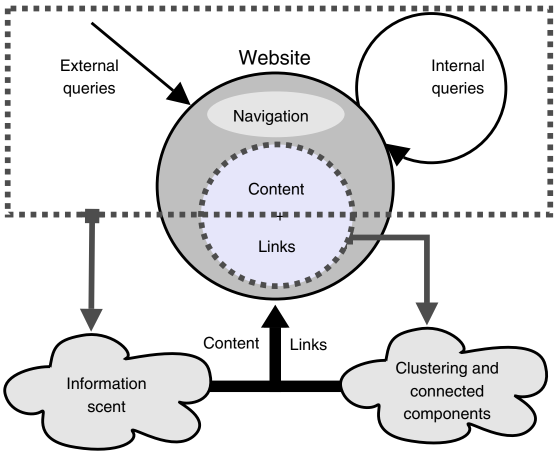

Fig. 2. Model description

3.2 Types of Queries

This model analyzes two different types of queries, which can be found in a web site's access registries, shown in figure 2. These queries are defined as follows:

External queries: These are queries submitted on Web search engines, after which users selected and visited documents in the web site.

Internal queries: These are queries submitted to a web site's internal search box. Additionally, external queries that are specified by the users for a particular site, will be considered as internal queries for that site.

For each query that is submitted in a search engine, a page with results is generated. This page has links to documents that the search engine considers appropriate for the query. The user can choose to visit zero or more documents from the results page, by reviewing the brief abstract of each document displayed, which allows the user to decide roughly if a document is a good match for her/his query.

By analyzing the users navigation within a web site, we can determine different types of pages. We divide the pages in two types: *documents reached without a search* (DRWS) and *documents reached only via queries* (DRQ). Note that DRWS and DRQ are not disjoint sets, because the same page can be reached using a search engine in one session, and without a search engine in another session. The important issue is to register how many times each of these different events occur. This classification is essential for discovering useful information from queries in a web site. Based on the different user actions we define five classes of queries as shown in table 2, where class B distinguishes queries associated to pages only visited by searching. Classes A and B can be further subdivided depending if they came from external or internal queries. All other classes are derived only from internal queries. Classes C' to E are classified either manually or with a thesaurus.

Table 2. Classes of queries

Class	Semantic exists	Answer	Clicks	Document type	Interest
A	yes	yes	yes	DRQ $\cap$ DRWS	low
B	yes	yes	yes	DRQ − DRWS	medium
C	yes	yes	no	—	low
C'	yes	no	no	—	medium
D	it should	no	no	—	high
E	no	no	no	—	none

We have built a prototype that does data cleansing, session and user identification, pattern discovery, query classification, separation of content and structure, and the text clustering based analysis. The data of the content and structure is extracted from the index generated by the web site's search engine crawler

Table 3. Examples for the different classes of queries

Class A	**Class B**	**Class C**	**Class C'**
admission tests	admission tests	scholarships	evening careers
universities	curriculum vitae	admission	diplomas
chat	bookstores	careers	Spain
employment	universities	sample tests	**Class D**
university scholarships	presentation letter	university chile	vocational test
admission exercises	English tests	law	scholarships Spain
thesis	university scholarships	admission results	compute score

[1], the tool used for the text clustering task was CLUTO [19], and the tool used for query pattern analysis was LPMINER [28]. The internal queries came from the internal search engine log and we considered only external queries from Google.

The prototype also includes a simple thesaurus, created from the user's feedback through a Web interface. This interface allows the user to group queries that have the same meaning. The thesaurus as well as the results of the manual classification of the queries C', D and E are saved permanently for each site, so the process does not need to be repeated for words previously classified.

As an example, Table 3 shows different examples taken from a one-week log from a portal targeted to university students. The log contained more than 56 thousand sessions, 355 thousand visited pages, 19 thousand external queries, and 4 thousand internal queries. On the left of this table, we show the most frequent queries for each class. Some suggestions that appear are to add information on scholarships in Spain or to provide a vocational test. Also, users do not like the answer for *law* or do not find information about night classes because the word *evening* is not used. Class E is not shown as many queries are not relevant for the portal (e.g. *msn* or *emotional intelligence*).

4 Query Clustering Framework

Following Wen *et al* [36], a *query session* consists of one query, a list of URLs, and the URLs the user clicked on. Figure 3 shows the relationships between the different entities that participate in the process induced by the use of a search engine. Our approach focuses in the relationship between queries, which will be defined using query traces, and the preferences/feedback of user about Web pages. The relationship between queries is obtained using the content of selected Web pages, which is indicated by the arrow from Web pages to queries.

The ranking and query recommender algorithms we present in this section consider only queries that appear in the query-log. Both algorithms follow ideas from a technique for building recommender systems called *collaborative filtering* [22]. Given a user searching for information, the idea is to first find similar users (via a k-neighborhood search or clustering process) and then suggest items preferred by the similar users to the current user. Since users are difficult to identify

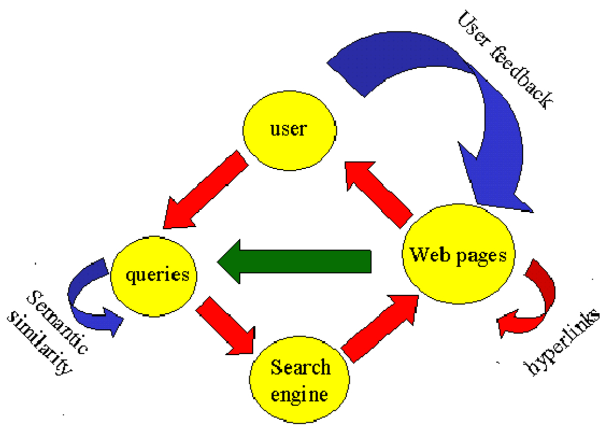

Fig. 3. Relationships of Entities in query logs

in search engines, we aggregate them in queries, i.e., sets of users searching for similar information. Thus the active user in our context is the input query. The items are respectively Web pages and queries in the ranking boosting and query recommendation algorithms.

Later we present experiments to evaluate the quality of our methods using a 15-day query-log from TodoCL. Currently the search engine has approximately 50,000 visits daily. The 15-day log contained over 6 thousand queries which have at least one click in their answers. There were over 22 thousand clicks in the answers, and these clicks were over 18,527 different URL's. The experiments consider the study of a set of 10 randomly selected queries. The 10 queries were selected following the probability distribution of the queries of the log.

4.1 Vector Representation of Query Traces and Query Similarity

In order to compute the similarity of two queries, we first build a term-weight vector for each query. Each term is weighted according to the number of occurrences and the number of clicks of the documents in which the term appears.

Given a query q, and a URL u, consider the popularity $\texttt{Pop}(u,q)$, that is, the number of clicks for page u when querying q. Let $\texttt{Tf}(t,u)$ be, as usual, the number of occurrences of term t in URL u. *Stopwords* are eliminated from the vocabulary considered. We define the *vector representation of a query trace*, $\boldsymbol{q}$, as follows:

$$\boldsymbol{q}[i] = \sum_{URL u} \frac{\texttt{Pop}(u,q) \times \texttt{Tf}(t_i,u)}{\max_t \texttt{Tf}(t,u)}$$

That is, Pop plays the role of *Idf* in the well-known tf-idf weighting scheme for the vector model.

We measure the similarity of two queries as the similarity of their trace vectors using the cosine function. Our notion of query similarity has several advantages. First it is simple and easy to compute. On the other hand, it allows to relate queries that happen to be worded differently but stem from the same information need. Therefore, semantic relationships of queries are captured. Another important advantage is that it avoids sparse similarity matrices which are usually generated using previous notions of query similarity.

4.2 Query Clustering

For the clustering process we used an implementation of a k-means algorithm, the CLUTO software package [43]. Each run of the algorithm computes k clusters, and to determine an adequate value of k we run the algorithm several times.

Many clusters represent clearly defined information needs of search engine users. Figure 4 shows details for three clusters, including the external and internal similarity. These examples, and many others found in our results, show the utility of our framework for discovering information needs related to queries.

Cluster Rank	ISim	ESim	Queries in Cluster	Descriptive keywords
84	0,697	0,015	office rental, rentals in Santiago, real state, apartment rental	office $(11,6\%)$, building $(7,5\%)$, real state $(5,9\%)$, real state agents $(4,2\%)$
252	0,447	0,007	car sales, cars Iquique, cars used, diesel, new cars,	cars $(49,4\%)$, used $(14,2\%)$, stock $(3,8\%)$, pickup truck $(3,7\%)$, jeep $(1,6\%)$
497	0,313	0,009	stamp, serigraph inputs, ink reload, cartridge	print $(11,4\%)$, ink $(7,3\%)$, stamping $(3,8\%)$, inkjet $(3,6\%)$

Fig. 4. Examples of clusters

4.3 Ranking Boosting

The algorithm operates in the following steps. Given an input query q, we find the cluster C to which the query belongs. Then the answer to q are the URL's that have been selected for the queries, ordered according to a rank score that considers two criteria: (a) the similarity of the page to the input query, it is measured using the notion of similarity based in equation 4.1. (b) the support or weight of the URL in the cluster. This is estimated as the popularity of the URL in the cluster. The rank score of a URL u is:

$$\texttt{Rank}(u) = \texttt{Sim}(q,u) \times \sum_{q_i \in C} \texttt{Pop}(u,q_i) \qquad (1)$$

This ranking can then be used to boost the original ranking algorithm, using a linear combination of the two ranks.

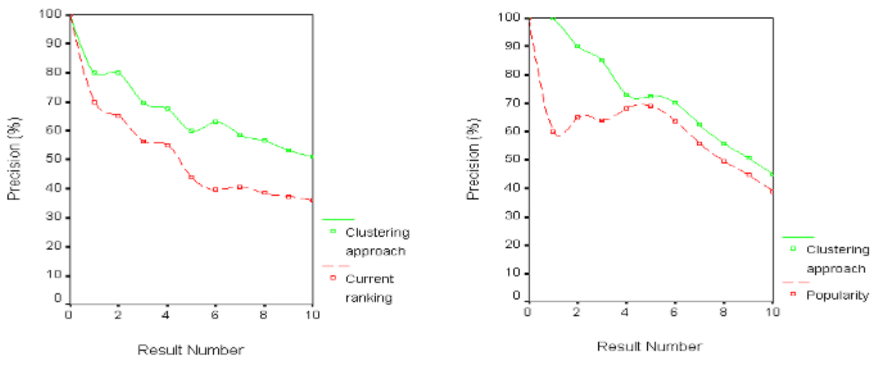

Fig. 5. Average retrieval precision of the proposed ranking algorithm (left) and the query commendation algorithm (right)

We compared our ranking algorithm with the algorithm provided by the search engine. The ranking algorithm of the search engine is based on a belief network which is trained with links and content of Web pages, and does not consider logs. Figure 5 (left) shows the average retrieval precision of the search engine and the proposed algorithm for ranking boosting. Only the top-10 answers of the queries studied are considered. The judgments of the relevance of the answers to each query were performed by different people. The graph shows that our algorithm can significantly boost the average precision of the search engine.

4.4 Query Recommendation

The query recommender algorithm is as follows. Given an *input query* q, again we first find the cluster C to which q belongs. Then we compute a rank score for each query in the cluster. The rank score of each query measures the *interest* of the query to users that submitted the input query. Finally, the related queries are returned ordered according to their rank score. The rank score of a query is based on notions of similarity to the input query and support of the query in the cluster. One may consider the number of times the query has been submitted as the support of a query. However, by analyzing the logs in our experiments we found popular queries whose answers are of little interest to users. In order to avoid this problem we compute the rank score of a query by aggregating the popularity and similarity (to the input query) of the URL's in the answer of the query. Let q be the input query, and let q_i be a query in the cluster C, and let U be the set of URL's in C, then the rank of q_i is:

$$\texttt{Rank}(q_i) = \sum_{u \in U} \texttt{Sim}(q, u) \times \texttt{Pop}(u, q_i) \tag{2}$$

In order to asses the quality of the query recommendation algorithm, we follow a similar approach to Fonseca *et al* [16]. The relevance of each query to the

Q	Ranking	Popularity
dress bride	house of bride	2
	dress wedding	7
	dress bridegroom	6
	wedding cake	3
	wedding rings	4
summer rental	rental apartments viña del mar owners	2
	rental apartments viña del mar	10
	viel properties	4
	rental house viña del mar	2

Fig. 6. Ranking of recommended queries for the input query dress bride

input query were judged by several people. They analyzed if the answers of the queries are of interest to the input query. Figure 6 shows the ranking of recommended queries for the input query *dress bride*. Notice that our algorithm finds queries with related terms, such as *wedding cake* and *bride dress*. Users looking for bride dresses are also interested in wedding cakes. In the other example we find that *Viña del Mar* is the top choice in Chile.

Figure 5 (right) shows the average precision for the queries considered in the experiments. We show precision vs. numbers of recommended queries. In average, we obtain a precision of 70% for the first six recommended queries. Therefore, the suggested queries are relevant to users that submitted the original queries. Our results also show that the rank scheme proposed is much better than the score obtained by considering only the popularity of the queries in the cluster.

5 Concluding Remarks

Our Web mining tool discovers in a simple way interesting words, by visualizing content that is relevant or not for the site. For example, class D queries may represent key missing content, products or services in a web site. Notice that the classification phase might be a drawback at the beginning, but in our own experience, it is almost negligible in the long run, as new queries seldom appear. Further study to evaluate the quality of the results is needed.

We have shown a clustering framework that allows to find groups of semantically related queries. Our experimental results show that our query clustering approach presents good results in two applications for improving a real vertical search engine. We have recently extended these results by adding techniques to unbias the distribution of clicks.

Traditional techniques for document retrieval can be used to handle queries in our framework. As an example, we could implement an inverted index scheme for terms in query traces to efficiently retrieve related queries. Such a scheme could also be used to recommend queries to input queries which do not appear in the logs.

Other measures for the interest of the queries in query recommendation are possible. For example, finding queries that share words but not clicked URL's.

This might imply that the common words have different meanings if the text of the URL's also are not shared. Hence we can detect polysemic words. On the other hand, if words are not shared and the many terms in the URL's are shared, that may imply a semantic relation among words that can be stored in an pseudo-ontology.

We are currently doing additional research in Web query mining to understand queries, find different type of users, as well as using queries to focus Web crawlers. For example, if we represent a given interest by a query vector Q using the vector model [5], a crawler can try to maximize the similarity of a retrieved vector page p with Q. We can extend the idea by representing all past queries in a search engine as a vector Q_t, which is updated using a time based average, with the last queries q by using a moving average: $Q_{t+1} = \alpha Q_t + (1-\alpha)q$, where α weights past versus current queries.

References

1. Akwan Information Technologies. Myweb search. `http://www.akwan.com.br`.
2. R. Baeza-Yates. Excavando la web (mining the web, original in Spanish). *El profesional de la información (The Information Professional)*, 13(1):4–10, Jan-Feb 2004.
3. R. Baeza-Yates, C. Hurtado, and M. Mendoza. Ranking boosting based in query clustering. In *Atlantic Web Intelligence Conference*, Cancun, Mexico, 2004. LNCS Springer.
4. R. Baeza-Yates, C. Hurtado, and M. Mendoza. Query Recommendation Using Query Logs in Search Engines. In *Workshop in Web Clustering*, Greece, 2004. Current Trends in Database Technology - EDBT 2004 Workshops, LNCS 3268, Springer.
5. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley & ACM Press, Harlow, UK, 1999.
6. Ricardo Baeza-Yates and Carlos Castillo. Relating web structure and user search behavior (extended poster). In *10th World Wide Web Conference*, Hong Kong, China, May 2001.
7. R. Baeza-Yates and F. Saint-Jean. A three level search engine index based in query log distribution. In *Proceedings of SPIRE 2003*, LNCS, Manaus, Brazil, October 2003. Springer.
8. Ricardo Baeza-Yates. Query Usage Mining in Search Engines. In *Web Mining: Applications and Techniques*, Anthony Scime, editor. Idea Group, 2004, 307–321.
9. R. Baeza-Yates and B. Poblete. A Web Usage and Content Mining Tool Centered in Queries, 2004, submitted.
10. P. Batista and M. J. Silva. Mining on-line newspaper web access logs, RPEC2-Workshop on recommendation and personalization on e-commerce, Spain 2002.
11. D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *KDD 2000*, Boston, MA, USA, 2000, 407–416.
12. R. Cooley, P. Tan, and J. Srivastava. Websift: the web site information filter system, 1999.
13. B. D. Davison, D. G. Deschenes, and D. B. Lewanda. Finding relevant website queries. In *Poster Proceedings of the Twelfth International World Wide Web Conference*, Budapest, Hungary, May 2003.

14. C. Ding and C. Chi. Towards an adaptive and task-specific ranking mechanism in web searching (poster session). In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 375–376, Athens, Greece, 2000. ACM Press. http://doi.acm.org/10.1145/345508.345663.
15. DirectHit: Main Page. `http://www.directhit.com`, 1998.
16. B. M. Fonseca, P. B. Golgher, E. S. De Moura, and N. Ziviani. Using association rules to discovery search engines related queries. In *First Latin American Web Congress (LA-WEB'03)*, November, 2003. Santiago, Chile.
17. Christoph Hlscher, and Gerhard Strube. Web Search Behavior of Internet Experts and Newbies, WWW9, Amsterdam, Netherlands, May 15 - 19, 2000.
18. Z. Huang, J. Ng, D. Cheung, M. Ng, and W. Ching. A cube model for web access sessions and cluster analysis, 2001.
19. G. Karypis. CLUTO, a clustering toolkit. Technical Report 02-017, Dept. of Computer Science, University of Minnesota, 2002. Available at http://www.cs.umn.edu/˜cluto.
20. Evangelos P. Markatos. On Caching Search Engine Query Results. In Proceedings of the 5th International Web Caching and Content Delivery Workshop, May 2000.
21. F. Masseglia, P. Poncelet, and M. Teisseire. Using data mining techniques on web access logs to dynamically improve hypertext structure, 1999.
22. M. Oconnor and J. Herlocker. Clustering items for collaborative filtering. Technical report, University of Minnesota, Minneapolis, MN, 1999. http://www.cs.umbc.edu/ ian/sigir99-rec/papers.
23. J. Pei, J. Han, B. Mortazavi-asl, and H. Zhu. Mining access patterns efficiently from web logs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 396–407, 2000.
24. Peter Pirolli. Computational Models of Information Scent-Following in a Very Large Browsable Text Collection, In Human Factors in Computing Systems: Proceedings of the CHI '97 Conference. ACM Press, New York, 3-10, 1997.
25. Iko Pramudiono, Takahiko Shintani, Katsumi Takahashi, and Masaru Kitsuregawa. User Behavior Analysis of Location Aware Search Engine, Mobile Data Management, 139-145, 2002.
26. Patricia Correia Saraiva, Edleno Silva de Moura, Nivio Ziviani, Wagner Meira, Rodrigo Fonseca, and Berthier Ribeiro-Neto. Rank-preserving two-level caching for scalable search engines, In Proceedings of the 24th annual international ACM Conference on Research and Development in Information Retrieval, New Orleans, USA, 51-58, September 2001.
27. Andreas Schaale, Carsten Wulf-Mathies, and Sonke Lieberam-Schmidt. A new approach to relevancy in Internet searching - the SVox Populi Algorithm T, arXiv.org e-Print archive, August 2003.
28. M. Seno and G. Karypis. LPMINER: An algorithm for finding frequent itemsets using length-decreasing support constraint. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 505–512. IEEE Computer Society, 2001.
29. C. Silverstein, M. Henzinger, M. Hannes, and M. Moricz. Analysis of a very large alta vista query log. In *SIGIR Forum*, pages 6–12, 1999. 33(3).
30. M. Spiliopoulou and L. C. Faulstich. WUM: a Web Utilization Miner. In *Workshop on the Web and Data Bases (WebDB98)*, pages 109–115, 1998.
31. M. Spiliopoulou, C. Pohle, and L. Faulstich. Improving the effectiveness of a web site with web usage mining. In *WEBKDD*, pages 142–162, 1999.

32. Amanda Spink, Dietmar Wolfram, Bernard J. Jansen, and Tefko Saracevic. Searching the Web: the public and their queries. Journal of the American Society for Information Science and Technology, 52(3), 226-234, 2001.
33. Amanda Spink, Bernard J. Jansen, Dietmar Wolfram, and Tefko Saracevic. From E-Sex to E-Commerce: Web Search Changes. *IEEE Computer* 35(3): 107-109, 2002.
34. Amanda Spink, Seda Ozmutlu, Huseyin C. Ozmutlu, and Bernard J. Jansen. U.S. Versus European Web Searching Trends. SIGIR Forum 26(2), 2002.
35. Todocl - Todo Chile en Internet. http://www.todocl.cl/, 2002.
36. J. Wen, J. Mie, and H. Zhang. Clustering user queries of a search engine. In *Proc. at 10th International World Wide Web Conference*. W3C, 2001.
37. Dietmar Wolfram. A Query-Level Examination of End User Searching Behaviour on the Excite Search Engine. Proceedings of the 28th Annual Conference Canadian Association for Information Science, 2000.
38. Y. Xie, and D. O'Hallaron, Locality in Search Engine Queries and Its Implications for Caching, Infocom 2002.
39. J. Xu and W. B. Croft. Improving the effectiveness of information retrieval with the local context analysis. *ACM Transaction of Information Systems*, 1(18):79–112, 2000.
40. G-R. Xue, H-J. Zeng, Z. Chen, W-Y. Ma, and C-J. Lu. Log Mining to Improve the Performance of Site Search, 1st Int. Workshop for Enhanced Web Search (MEWS 2002), Singapore, Dec 2002, IEEE CS Press, 238-245.
41. O. R. Zaiane and A. Strilets. Finding similar queries to satisfy searches based on query traces. In *Proceedings of the International Workshop on Efficient Web-Based Information Systems (EWIS)*, Montpellier, France, September, 2002.
42. D. Zhang and Y. Dong. A novel web usage mining approach for search engines. *Computer Networks* 39(3): 303-310, April 2002.
43. Y. Zhao and G. Karypis. Comparison of agglomerative and partitional document clustering algorithms. In *SIAM Workshop on Clustering High-dimensional Data and its Applications*, 2002.