www.ijecs.in

International Journal Of Engineering And Computer Science ISSN:2319-7242

Volume 4 Issue 7 July 2015, Page No. 13144-13150

Design of Query Recommendation System using Clustering & Rank Updater

Priyanka Rani, Mrs. Annu Mor

1. **Priyanka Rani**, M.Tech Scholar, *Computer Science Engineering Department*
SGT Institute of Engineering & Technology, Gurgaon
Parigtm27@gmail.com

2 **Annu Mor**, *Associate Professor, Computer Science Engineering Department*
SGT Institute of Engineering & Technology, Gurgaon
Annu.mor14@gmail.com

ABSTRACT

In this paper I propose a method that, given a query submitted to a search engine, suggests a list of related queries. Query recommendation is a method to improve search results in web. This paper presents a method for mining search engine query logs to obtain fast query recommendation on a large scale. Search engines generally return long list of ranked pages, finding the important information related to a particular topic is becoming increasingly difficult and therefore, optimized search engines become one of the most popular solution available. In this work, an algorithm has been applied to recommend related queries to a query submitted by user. For this, the technology used for allowing query recommendations is query log which contains attributes like query name, clicked URL, rank, time. Then, the similarity based on keywords as well as clicked URL's is calculated. Additionally, clusters have been obtained by combining the similarities of both keywords and clicked URL's. The related queries are based in previously issued queries The method not only discovers the related queries, but also ranks them according to a relevance criterion. In this paper the rank is updated only the clicked URL, not all the related URL's of the page.

Keywords:- Query Log, Search Engine, and Query Clustering, Query Similarity, Information Retrieval, Page Rank Updater.

1. INTRODUCTION

A key factor for the popularity of today's Web search engines is the frienkly user interfaces to provide. Indeed, search engines allow users to specify queries simply as lists of keywords, following the approach of traditional information retrieval systems. Keywords may refer to broad topics, to technical terminology, or even to proper nouns that can be used to guide the search process to the relevant collection of documents. Despite that this simple interaction mechanism has proved to be successful for searching the Web, a list of keywords is not always a good descriptor of the information needs of users. It is not always easy for users to formulate effective queries to search engines. One reason for this is the ambiguity that arises in many terms of a language. Queries having ambiguous terms may retrieve documents which are not what users are searching for. On the other hand, users typically submit very short queries to the search engine, and short queries are more likely to be ambiguous. From a study of the log of a Popular search engine, it concludes that most queries are short and imprecise. Users searching for the same information may phrase their queries differently. Often, users try different queries until they are satisfied with the results. 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 aboutthe information they are searching, and worst, they could not even be certain about what to search for. The idea is to use these expert queries to help non-expert users. In order to overcome these problems, some search engines have
Their aim is to help the users to specify alternative related queries in their search process. Typically, the list of suggested queries is computed by processing the query log of the search engine, which stores the history of previously submitted queries and the URL's selected in their answers. But it is one main fact that there may be more than one page for the related query and they also shows almost the same thing, then what to do?

II.PRIOR APPROACH

The propose work is 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.

The notion of query recommendation has been a subject of interest since many years. A number of

researchers have discussed the problem of finding relevant search results from the search engines. Relevant query recommendation research is mainly based on previous query log of the search engine, which contains the history of submitted query and the user selected URLs. Bee ferman and Berger[1] exploited "click through data" in clustering URLs and queries using graph-based iterative cluster-ing technique. Both of their algorithms are difficult to deal with in practice due to query log sparseness. That is to say, only a part of popular queries have sufficient log information for mining their common clicked URLs while distance matrices between most queries from real query logs are very sparse. As a result, many queries with semantic similarity might appear orthogonal in such matrices. However, the fact that similar queries are submitted by different users in most of case, will also lead to serious problem. 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. Query expansion is also adopted by search engines to recommend related queries. Its idea is to reformulate the query such that it gets closer to the term weight vector space of the documents the user is looking for. This approach aims at construction of queries rather than recommend previous registered queries in real log However, a critical look at the available literature indicates that from very beginning, search engines are using some kind of optimization on their search results but they are not much beneficial due to the problems of finding the required information within search results. Hence, a mechanism needs to be introduced gives prime importance to the information needs of users. Query log that keeps record of user queries on the basis of occurrence of query in the query cluster which is formed by clustering similar queries one of their top ten results. These queries are then used as suggestions. Meiet al. [3] proposed an algorithm based on hitting time on the Query-URL bipartite graph derived from search logs. Starting from a given initial query, a sub graph is extracted from the Query-URL bipartite using depth first search. A random walk is then conducted on this sub graph and hitting time is computed for all the query nodes. Queries with the smallest hitting time are then used as suggestions. Neelam Duhan, A. K. Sharma [4] pre-mines the query logs to retrieve the potential clusters of queries and then finds the most popular queries in each cluster. Each cluster entries are mined to extract sequential patterns of pages accessed by the users. The outputs of mining processes are utilized to return relevant results with popular historical queries. Yang Song et al. [5] proposed to suggest queries using Term-Transition graphs in which a large amount of user preference data is mined from query logs. It constructs a term preference graph where each node is a term in the query and each directed edge a preference. Then a topic-biased Page Rank model is trained for each of the query and it guides the decision of expanding relevant terms to the original query, removing terms from the original query, or replacing existing terms with relevant terms..

. Query clustering helps to find appropriate terms for this expansion.

III. PURPOSED WORK

METHODOLOGY

When user submits a query(Fig. 1) on the interface of search engine, the query processor component matches the query terms with the index repository of the search engine and taking a list of matched documents in reply. On the reverse order, result optimization system performs its task of gathering user intentions from the query logs. The user browsing behavior as well as the submitted queries and clicked URLs get stored in the logs and are analyzed continuously by the Similarity Analyzer module, the output of which is forwarded to the Query Clustering Tool to create potential groups of queries based on their similarities. Then the clusters are stored in query cluster database. Then the favored query finder find out the relevant query from the database. The query recommender recommends the similar query. The Rank Updater component takes as input the matched documents retrieved by query processor. It improves the ranks of pages to all according to sequential patterns with optimized rank get stored in the interface of search engine which produces final results to user.. The working for different functional modules are explained below in the next subsections.

The proposed system works in the following steps

1. Query Log
2. Similarity Analyzer
3. Query Clustering Tool
4. Favored Query Finder
5. Rank Updater
6. Query Recommender

1. Query Logs

sQuery log has been a popular data source for query recommendation. Query logs are repositories that record all the interactions of users with a search engine for gaining insight into how a search engine is used and what the users' interests are. Since they form a complete record of what users searched for in a given time frame. Depending on the specifics of how the data is collected,

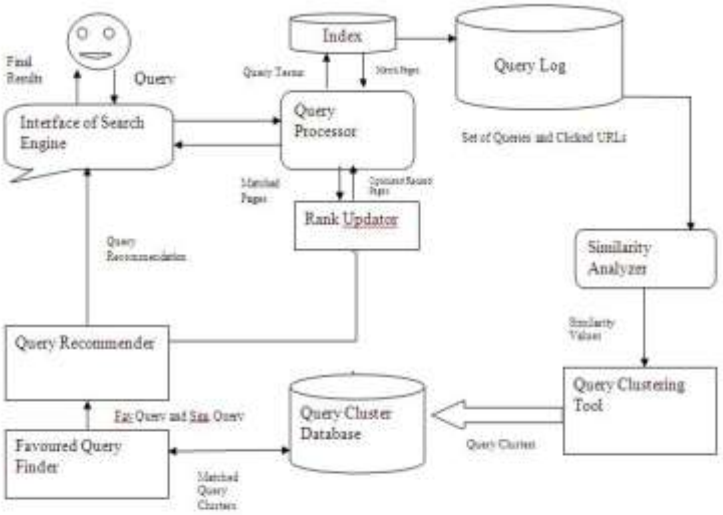


Fig.1

The approach taken by this module is based on two criteria:

a). one is on the queries keywords, and

b). the other on clicked URLs

typically logs of search engines include the following entries:

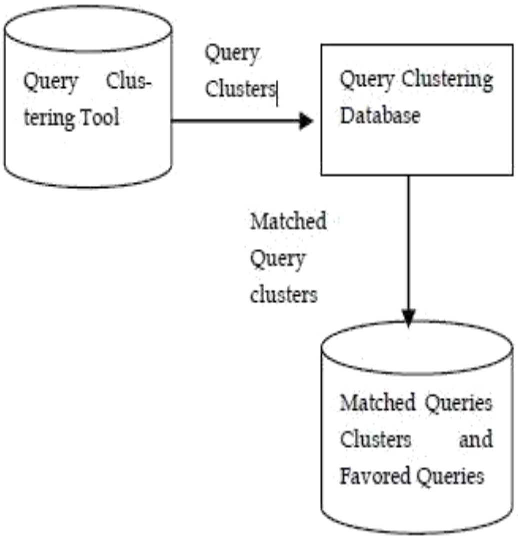
1. User IDs,
2. Query q issued by the user,
3. URL selected by the user
4. Rank of the URL clicked for the query
5 .Time at which the query has been submitted for search.

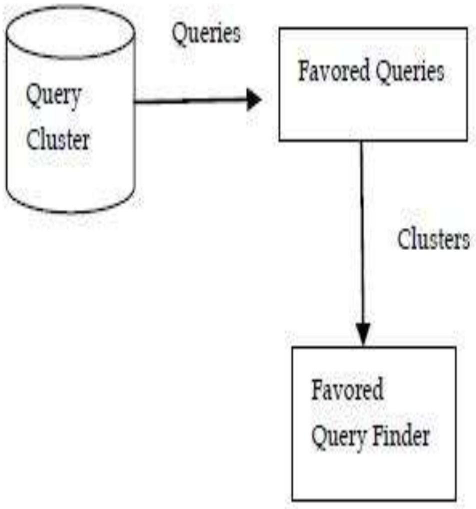
2. Similarity Analyzer

The next step in proposed system is computing the query similarity. It is an important crisis and has a wide range of applications in Information Retrieval in query recommendation. Traditional approaches make the use of keywords extracted from documents. If two documents share some keywords, then they are thought to be similar to some extent. The more they share common keywords, and the more these common keywords are important, the

3. Query Clustering Tool

In support of the clustering process, this tool is used to cluster user queries using query clustering tool built by search engines and for this it assigns query cluster database log entries, which in result produces matched query clusters and favored queries as shown in Fig


Query Clustering Tool
Query Clusters
Query Clustering Database
Matched Query clusters
Matched Queries Clusters and Favored Queries



An important component in this work is the concept of clustering queries in user logs. The query clustering is a preprocessing phase and it can be conducted at periodical and regular intervals. Even though the need for query clustering is somewhat new, there have been general studies on document clustering, which are similar to query clustering. However, it is not reasonable to easily apply any document clustering algorithms to queries due to their own characteristics. It is usually observed that queries submitted to the search engines typically are very short, so the clustering algorithm should be suitable for short texts. Additionally query logs are usually very large, the method should be able of handling a large data set in reasonable time and space constraints. Furthermore, due to the fact that the log data changes daily, the method should also be incremental.

4. Favored Query Finder

When query [9] clusters are formed, another phase is to find a set of favored queries from each cluster. Query is said to be favored query that occupies the foremost portion of the search requests in a cluster. The process of finding favored queries is shown in fig4 which find the favored queries in one cluster. The method is applied in every the clusters and output is stored in the Query Cluster Database

5. Rank Updater

Two popular algorithms were introduced in 1998 to rank web pages by popularity and provide better search results. They are:

•HITS (Hypertext Induced Topic
Search) •Page Rank

HITS was proposed by Jon Kleinberg who was a young scientist at IBM in Silicon Valley and now a professor at Cornell University.

Page Rank was proposed by Sergey Brin and Larry Page, students at Stanford University and the founders of Google. The Web's hyperlink structure forms a massive directed graph.

Hyperlinks into a page are called <u>in link</u> and point into nodes and <u>out links</u> point out from nodes.

Page Rank is a numeric value that represents the importance of a page present on the web.

When one page links to another page, it is effectively casting a vote for the other page. More votes implies more importance. Importance of the page that is casting the vote determines the importance of the vote. A web page is important if it is pointed to by other important web pages.

Google calculates a page's importance from the votes cast for it. Importance of each vote is taken into account when a page's Page Rank is calculated. Page Rank is Google's way of deciding a page's importance. It matters because it is one of the factors that determine a page's ranking in the search results.

6. Query Recommender

Query Recommender provides the user with a set of queries which are recommended with the most popular query. The recommended queries are those that are related to the query submitted by the user and therefore these queries are contained in the cluster of NOKIA PHONE

- NOKIA STORE
- NOKIA CARE
- NOKIA TAB

The recommended queries are sorted with popular query being highlighted. When user submits a query, its keywords are matched in Query cluster database and the queries in the matched cluster are outputted by the Query Recommender on the interface of search engine. The user can carry on with the same query otherwise can decide any one of the recommendation.

ALGORITHMS AND FORMULAS USED

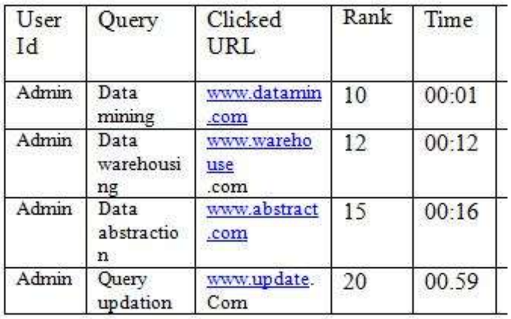
Query Logs

As we defined earlier that in the query log having these fields:-

1. User IDs,
2. Query q issued by the user,
3. URL selected by the user
4. Rank of the URL clicked for the query
5 .Time at which the query has been submitted for search.

Then this table formed:

Table 1:- Illustration of Query Log



User Id	Query	Clicked URL	Rank	Time
Admin	Data mining	www.datamin.com	10	00:01
Admin	Data warehousing	www.warehouse .com	12	00:12
Admin	Data abstraction	www.abstract.com	15	00:16
Admin	Query updation	www.update. Com	20	00.59

Query Similarity

The approach taken by this module is based on two criteria: one is on the queries keywords, and the other on clicked URLs. These approaches are formulated below:

Similarity based on query keywords

If two user queries contain the same or similar terms, they denote the same or similar information needs. The following formula is used to measure the content similarity between two queries.

$$\mathrm{Sim}(p, q) = \frac{|KW(p, q)|}{|kw(p)\ U\ kw(q)|}$$

Where kw (p) and kw (q) are the sets of keywords in the queries p and q respectively, KW (p, q) is the set of common keywords in two queries

Similarity Based On Clicked URLs

The following formula dictates the similarity function based on documents clicks.

$$\mathrm{Sim}_{clickURL}(p,q) = \frac{\sum LC(p, di) + LC(q, di)}{\sum LC(p, xi) + LC(q, xi)}$$

Where LC (p, d) and LC (q, d) are the number of clicks on document d corresponding to queries p and q respectively. CD (p) and CD (q) are the sets of clicked documents corresponding to queries p and q respectively.

Combined Similarity Measure

It is better to combine them in a single measure. A simple way to do it is to combine both measures linearly as follows:

$$\mathrm{Sim}_{combines}(p,q) = \alpha. \mathrm{Sim}_{Keyword}(p,q) + \beta. \mathrm{Sim}_{clickURL}(p,q)$$

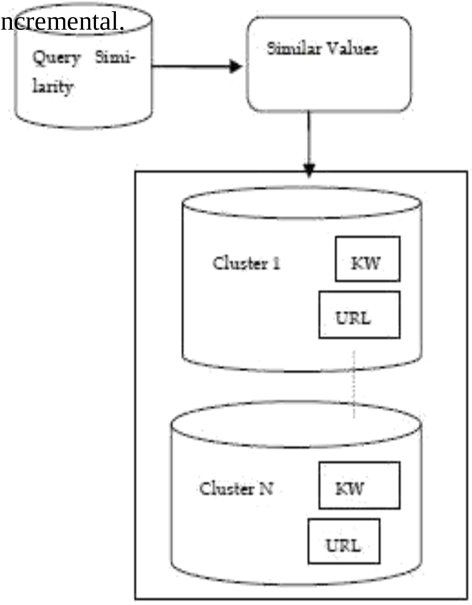
Where α and β are constants with $0<=\alpha(\text{and } \beta)<=1$ and $\alpha+\beta=1$

The values of constants can be decided by the expert analysts depending on the importance being given to two similarity measures. In the current implementation, these parameters are taken to be 0.5 each.

CLUSTERING ALGORITHM

Another question involved is the clustering algorithm proper. There are many clustering algorithms available to us. The main characteristics that guide our choice are the following ones:

The algorithm should not require manual setting of the resulting form of the clusters, e.g. the number of clusters. It is unreasonable to determine these parameters manually in advance.Since we only want to find FAQs, the algorithm should filter out those queries with low frequencies. Since query logs usually are very large, the algorithm should be capable of handling a large data set within reasonable time and space constraints.Due to the fact that the log data changes daily, the algorithm should be incremental.



***Algorithm**:* Query_Clustering(Q,α,β,τ)

Given : A set of n queries and corresponding clicked url'sstored in an array

Q[q1,URL1…..URL m] . 1<=i<=n
α=β=0.5

Similarity Threshold τ
Output : Aset C={C1,C2….Ck} of k query clusters
//Start Algorithm

```
K=1;              // k is the number of clusters

For (each query p in Q)
Set ClusterId(p) - Null;
 //Initially No Cluster is clustered
For (each p € Q)
{ClusterId(p) = Ck; Ck –{ p };

For each q € Q such that p ≠ q {
$\mathrm{Sim}(p, q) = \frac{|KW(p, q)|}{|kw(p)\ U\ kw(q)|}$
  For
$\mathrm{Sim}_{clickURL}(p,q) = \frac{\sum LC(p,di)+LC(q,di)}{\sum LC(p,xi)+LC(q,xi)}$
If($\mathrm{Sim}_{combines}(p,q) > \tau$)
  Set ClusterId(q) = Ck;
  Ck = Ck U {k};
  Else
  Continue;
  }  //  End
  K=K+1;

  } //End Outer For
  Return Query Cluster Set C;
```

Table 2 – Query log for query clustering

SRN O.	QUERY	URL
1.	DATA WAREHOUSE	www.dmining.com
		www.google.com
2.	DATA MINING	www.datawarehousing .com
		www.google.com

FAVORED QUERY FINDER ALGORITHM

```
Algorithm: Favored Query
Finder() I/P : A Cluster of
Queries
O/P : True or
False. //Start of
Algorithm
1. Queries Which are exactly same club them
and make a set of the <query,IP addresses>
pairs.
2. For(each q € Clusters)
3. Calculate the weights of query as :
Wt = No. Of IP addresses which Fired the
query/Total No. of IP Addresses in that
cluster If (Wt >= threshold Value) then
Return True; //Query is considered as favored
query Else
Return False; //Query is considered as disfavored
```

FORMULA USED FOR RANK UPDATER

Where *Iu* and *Ip* represent the number of links of page *u* and page *p*, respectively. *R* (*v*) denotes the reference page list of page *v*. *Wout*(*v*,*u*) given in eq. (4) is the weight of *link* (*v*, *u*) calculated based on the number of outlinks of page *u* and the number of outlinks of all reference pages of page *v*.

$$W^{out}{}_{(v,u)} = \frac{O_u}{\sum_{p \in R(v)} O_p}$$

Where *Ou* and *Op* represent the number of outlinks of page *u* and page *p*, respectively. *R* (*v*) denotes the reference page list of page *v*.

D(v,u) = D (u)/D (v)

Here D(u) and D (p) are the no. of duplicates. Considering the importance of pages, the original PageRank formula is modified in eq. as

$$PR(u) = (1-d) + d \sum_{v \in B(u)} PR(v) W^{in}{}_{(v,u)} W^{out}{}_{(v,u)}$$

New Formula:

PR(u)=(1-d)+d ∑PR(v) *W(in)*W(out)*D

Table 3 :- Rank Optimization

Query page	Related URL	Previous Rank	New rank	
DATA MINING	www.google.c om	30	31	
	www.dmining.c om	28	28	

IV. CONCLUSION AND FUTURE WORK

In this paper, Architecture of result optimization system has been proposed based on query log for implementing effective web search. The most significant feature is that the result optimization method is based on users' feedback, which determines the relevance between URL's and user query words. The returned URL with better ranks are directly mapped to the user feedbacks and dictate higher relevance than URL that exist in the result list but are never accessed by the user. Hence, the time user spends for looking for the required information from search result list can be reduced and the more important URL can be presented. As the system based on click through data in query log and semantic search has been proposed for implementing effective web search, the most important feature is that the proposed approach is based on users' behavior, which determines the relevance between URL and user query words. The results obtained from practical evaluation are quite effective in respect to reduced search space and enhanced the use of interactive web search engines. As the future work, we can apply a more relevant formulas and algorithms to update the query more efficiently. Although a conclusion may review the main points of the paper, do not repeat the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions. And we prove it that the work done by this paper is reached to its main point of designation to revive.

REFERENCES

[1].D.Beeferman and A. Berger. Agglomerative Clustering of a Search Engine Query log. In KDD, pages 407-416, Boston, MA USA, 2000.

[2].T.J. Berners-Lee, R. Cailliau, J-F Groff, B. Pollermann, CERN, "World-Wide Web: The Information Universe", published in Electronic Networking: Research, Applications and Policy, Vol. 2 No 1, Spring 1992, Meckler Publishing, Westport, CT, USA..

[3] Neelam Duhan, A. K. Sharma. “Rank Optimization and Query Recommendation in Search Engines using Web Log Mining Techniques”. In journal of computing, volume 2, issue 12, December 2010.

[4]Ricardo Baeza-Yates1, Carlos Hurtado1, and Marcelo Mendoza Query Recommendation usingQuery Logs in Search Engines

[5]Mayank Arora1 and Neelam Duhan2 “Design of Query Suggestion System using Search Logs and Query Semantics

[6] Rachna Chaudhary, Nikita Taneja ” A novel approach for Query Recommendation Via query logs”

[7]Rachna Chaudhary, Nikita Taneja ”Query Recommendation for Optimizing the Search Engine Results”