Time Independent Query Recommendations Using Concept Based User Profile from Search Engine Query Logs

R. Umagandhi and A.V. Enthil Kumar

Abstract: Search engines are highly confided resources of the people in mustering web information or obtaining relevant data approached for. The query log file contains an entry for every request posed by the user to the search engine and it is maintained in the system desktop or in the proxy server. Query log mining improves the performance of the search engine. The proposed algorithm mines the query log file which discovers the similar query keywords, URLs and the concepts based on both positive and negative preferences in its first phase. In the Second phase, the query cluster and the URL cluster is generated by using the combined similarity measure generated from the first phase. The cluster recommends the query to the user to frame their future queries based on their previous search histories and click through data. Expert’s Query keyword and feedback are also considered for providing the recommendations. This approach also recommends the URLs to the user to be selected for their future queries.

Key words: Similarity • Query term • Search Engine • Query Cluster • URL

INTRODUCTION

Search engine plays an important role in the Information Retrieval process to provide the requested information based on the query keywords in terms of web snippets. The results retrieved by the search engine is not relevant all the times, sometimes irrelevant URLs are also recommended to the user because of shorter and ambiguous query keywords. A study done by C. Silverstein on Alta Vista Query Log has shown that more than 85% of queries contain less than three terms and the average length of the queries are 2.35 terms [1]. Thus, the shorter length queries do not provide any meaningful, relevant and needed information to the users.

Query recommendations instigate an authentic search trusted for its precision in procuring appropriate and relevant results for each search through the search engine. The methodical recommendations enumerated to the web user in ordaining meaningful queries, as a matter of fact, helps to retrieve relevant and accurate results during every search. These suggestions depend on the actual interest of the user. This indeed, is observed through the past search histories. For instance, a user’s submission of the query term ‘java’ to the search process shall be considered. In this search, the user is perceived to be really interested in ‘java bike’ rather than the ‘java language’. Here, the user’s absolute determination to retrieve information about two-wheeler vehicles, that too exclusively, ‘java bike’, of course may admirably be satisfied by the query recommendation system that supplies potential suggestions about ‘java bike’ also during the triggering of the same query, ‘java’.

Another example where the user, $U_1$ seeks information for ‘cloud computing’ with a circuitous query, while the user, $U_2$ explores the same with an apposite query may perhaps be discussed. In this case, the user $U_1$ attempts to obtain information on ‘cloud computing’ in lack of appropriate knowledge about the query terminology. Hence, $U_1$ tries the search with the keyword ‘network computing’ instead of ‘cloud computing’. This effort fetches top documents that do not match with the required information. In this cumbersome process of exploration $U_1$ arrives at the appropriate URL after consuming a long time. However, $U_2$ explores with the appropriate query terminology, ‘cloud computing’ itself to retrieve relevant information from required URL of user $U_2$.
in a brief span of time. In such circumstances, to reduce the burden of a circuitous navigation of the users, either
due to query terms bearing semantic ambiguity or due to
feeding of inappropriate query terms in the search
engine, an algorithm is proposed as part of this research.
The prime function of this algorithm is generating query
clusters. As the intention of the users $U_i$ and $U_j$ are
observed to be the same and they are clustered.
These clusters contain the query keywords and the URL
click history of smart user $U_i$ and an average user $U_j$.
Further, this cluster is used to give recommendations for
the first user $U_i$ by using the query keywords of $U_j$.

The rest of the paper is organised as follows: Section
2 reviews the related work. Section 3 defines the
generation of query clusters which are used in the
recommendation process. Section 4 discusses the
experiments and results. Finally the paper is concluded in
Section 5.

**Related Works:** The content based approach suggests a
similarity measure between queries according to the
keywords given by the user [2]. Here queries are grouped
into the same cluster if they contain one or more common
terms. Different types of similarity measures like Cosine
similarity, Jaccard similarity [3] etc. are also available to
find the content based similarities. The similarity between
the queries is determined by calculating the overlap in
documents returned by the queries [4]. Various query
similarity techniques are described in [5-7]. Similarities
based on the Query Keywords and Cosine similarity are
combined in [8]. The cross reference similarity and
keyword similarity are combined in [7] to generate the
query cluster. Query recommendation system is also
based on Semantic of the query keyword [9]. Doug
Beeferman et.al [10] generates the cluster of similar
queries and URL and it is content ignorant. Qiankun
Zhao et al [11] generated the query cluster based on the
time period. Similar queries occurs at the particular time
period are considered only for the cluster formation. But
our proposed algorithm deals the similar queries occur at
any time interval.

Figure 1 shows an example for query
recommendations at the time of framing the queries in
Google search engine for the query ‘inheritance’. Figure 2 shows an example for query recommendations to
the frame future queries at the end of the first resultant
page given by the Google search engine for the same
query ‘inheritance’.

The recommendations are used to frame the queries
in future. Our algorithm frames the query cluster which
provides the query suggestions for the user to frame the
queries in future.

The proposed work combines the similar queries
occurs at any time period based on the similarity of query
keywords, clicked URLs and the concepts based on both
positive and negative preferences. The user scans the
search result from the top to the bottom and decided to
click the documents because the concepts available in the
clicked documents are required information for the user.
Those concepts apparently available in the clicked
documents are called *concepts with positive preferences*
whereas; those concepts existing in the un-clicked
documents are called *concepts with negative preferences*.
Table 1 juxtaposes the URLs along with their respective
concepts generated for the query ‘inheritance’. At this
point, the user is witnessed to have clicked the
documents D1, D6 and D8. As the user preferred the
concepts in {D1, D6, D8} these concepts are addressed as
positive preferences and the concepts in {D2, D3, D4, D5,
D7, D9, D10} are not preferred by the user hence, they are
Table 1: Clicked Documents with Concepts

<table>
<thead>
<tr>
<th>Document</th>
<th>URL</th>
<th>Concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td>v D1</td>
<td>en.wikipedia.org/wiki/Inheritance_(object-oriented-programming)</td>
<td>Object-oriented programming, inheritance</td>
</tr>
<tr>
<td>D2</td>
<td>en.wikipedia.org/wiki/Inheritance</td>
<td>Individual, characteristic</td>
</tr>
<tr>
<td>D3</td>
<td>docs.oracle.com/javase/tutorial/java/landisubclasses.html</td>
<td>New class</td>
</tr>
<tr>
<td>D4</td>
<td>docs.oracle.com/javase/tutorial/java/concepts/inheritance.html</td>
<td>Common, bikes</td>
</tr>
<tr>
<td>D5</td>
<td><a href="http://www.alagaesia.com">www.alagaesia.com</a></td>
<td>Trilogy</td>
</tr>
<tr>
<td>v D6</td>
<td><a href="http://www.tutorials%EF%BF%BD%EF%BF%BDcom/cpp_inheritance.htm">www.tutorials��com/cpp_inheritance.htm</a></td>
<td>beginner's tutorial, c++</td>
</tr>
<tr>
<td>D7</td>
<td><a href="http://www.inheritanceindia.co.in">www.inheritanceindia.co.in</a></td>
<td>Company, lands</td>
</tr>
<tr>
<td>v D8</td>
<td><a href="http://www.cppplus.com/doc/tutorial/inheritance">www.cppplus.com/doc/tutorial/inheritance</a></td>
<td>Friend, private, protected</td>
</tr>
<tr>
<td>D9</td>
<td>propelorm.org/documentation/09-inheritance.html</td>
<td>Table, structure, columns</td>
</tr>
<tr>
<td>D10</td>
<td>processing.org/learning/basics/inheritance.html</td>
<td>Class, object-oriented programming</td>
</tr>
</tbody>
</table>

Table 2: Attributes along with its Description

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AnonID</td>
<td>Anonymous ID assigned for every user</td>
</tr>
<tr>
<td>Query</td>
<td>The Query supplied by the user</td>
</tr>
<tr>
<td>QueryTime</td>
<td>The date and time on which the query triggered by the user</td>
</tr>
<tr>
<td>ItemRank</td>
<td>Rank assigned to each clicked URL</td>
</tr>
<tr>
<td>ClickURL</td>
<td>The URL address clicked by the user when the query was supplied</td>
</tr>
<tr>
<td>Concepts</td>
<td>Important concepts retrieved from the clicked URL</td>
</tr>
</tbody>
</table>

addressed as negative preferences. Here the Query Cluster is generated; it contains a set of similar queries triggered at any time used for recommending and framing the future query for different users.

**Generation of Query Cluster**

**Processing of Query Log File:** In order to identify the future queries, the log entries from the search engine’s query log are analysed. The search histories are organised based on the following attributes given in Table 2. The attributes are

< AnonID, Query, QueryTime, ItemRank, ClickURL, Concepts>

The search engine responds to the user’s query $Q$ in terms of web snippets. A “Web-snippet” denotes the title, summary and URL of a web page returned by the search engines. If a keyword exists frequently in the top documents of the web snippets then it is an important concept regarding that query. The prominent recurrence of a keyword in the top retrieved documents of the web snippets adds adequate significance to the concept corresponding to that particular query. Such recurrent concepts of the clicked web snippets are cautiously obtained through the support formula [12],

$$\text{Support}(C_i) = \frac{sf(C_i)}{n}$$  \hspace{1cm} (1)

Where $sf(C_i)$ is the number of web snippets containing $C_i$ and $n$ is the total number of web snippets. If the support of a concept $C_i$ is greater than the threshold $s$ then $C_i$ is an important concept for the query $q$. In these experiments performed for this research, the threshold value for the concept retrieval is 3. The concept occurs in minimum of 3 out of top 10 documents. The concept in clicked URL which satisfies the threshold value is treated as concepts with positive preferences. The concept satisfies the threshold value but occurs in un-clicked URL is considered as concept with negative preferences.

Table 3 depicts the sample log entries for the proposed work. From the log entries given in Table 3, 8 unique URLs and queries are retrieved and it is given in Table 4 and Table 5. For the unique URL and Query an ID is assigned.

**Architecture for Generation of Query Cluster:** The process of Query Cluster Generation is given in Figure 3. The user submits the query through the middleware which goes to the Search Engine. The user’s request and their navigational behaviours are recorded in the query log file. The user scans the search result from the top to the bottom and decides that the retrieved results are not relevant for their request. Sometimes the user scans the search result and satisfied about the information available in the abstract of the web snippets itself. For these cases the user will not click any URL, the message “NoClick” will be assigned to the attribute ClickURL. The log entries are cleaned, the users and their sessions are identified by using the algorithms given in [13, 14]. Different pre processing tools are available to pre process the log entries [15-17].

Once the log entries are cleaned, the unique queries submitted by the different users and their clicked unique URLs are identified and stored in the data files. In the First phase similarity of query terms, clicked URLs which contains the cross references are computed. Similarities between the concepts based on both positive and
negative preferences are also computed in the First phase. Next the query, URL and concept similarity measures are combined which are used to find and cluster the similar queries in the Second phase. The query cluster recommends different queries to the users for their future search. The similarities between the queries are obtained by combining Keyword, URL and Concept Similarity. The algorithm QCLUSTER in Section 3.3 generates the cluster of similar queries.

**Similarity Between Queries:** The clustering technique used here is obtained from the following observations. First, the user issues the queries which have some common keywords and clicked two different URLs. Here the queries are clustered on the basis of query similarity. Second, the user gives two different queries to search the same information and clicked the same URL. The queries are clustered on the basis of URL similarity. Next, the queries and URLs are different but the concepts appeared in the URL may be same. Now the queries are clustered by using concept similarity.

The similarity between the query terms and URLs are calculated using the equations (2) and (3). The numerator of our proposed work is similar to the numerator of Jaccard Coefficient [5].

\[
\text{Key word Similarity } \left(Q_i, Q_j\right) = \frac{\text{Number of Similar keywords in the queries } Q_i \text{ and } Q_j}{\text{Total number of keywords in } Q_i \text{ and } Q_j}
\]

\[
\text{Key word Similarity } \left(Q_i, Q_j\right) = \frac{\text{keywords } Q_i \text{ and } Q_j}{\text{keywords}(Q_i) + \text{keywords}(Q_j)}
\]

\[(2)\]

If the queries \(Q_i\) and \(Q_j\) share some common terms in their keywords then the queries are similar. The number of similar URLs clicked for \(Q_i\) and \(Q_j\) is calculated as

\[
\text{URL Similarity } \left(Q_i, Q_j\right) = \frac{\text{Number of Similar URLs clicked for } Q_i \text{ and } Q_j}{\text{Total number of URLs clicked for } Q_i \text{ and } Q_j}
\]

\[
\text{URL Similarity } \left(Q_i, Q_j\right) = \frac{\text{Count } \left[ \text{URL}(Q_i) \cap \text{URL}(Q_j) \right]}{\text{Count } \left[ \text{URL}(Q_i) \right] + \text{Count } \left[ \text{URL}(Q_j) \right]}
\]

\[(3)\]
The function count() is used to find the number of URLs clicked for the given query. The URL count is calculated by using the algorithm HASHURLCOUNT [18].

Query cluster based on the similarity of query terms and URL does not provide the accurate cluster. Since the queries and URLs are dissimilar, but the concepts retrieved from the different URLs may be similar. The proposed work concentrates the similarity on concepts based on both positive and negative preferences.

\[
\text{Concept Similarity} \left( Q_i, Q_j \right) = \frac{\text{Number of Positive concepts available for } Q_i \text{ and } Q_j}{\text{count } \left( \text{posconcept} \left( Q_i \right) \cap \text{posconcept} \left( Q_j \right) \right)} + \frac{\text{Number of Negative concepts available for } Q_i \text{ and } Q_j}{\text{count } \left( \text{negconcept} \left( Q_i \right) \cap \text{negconcept} \left( Q_j \right) \right)}
\]

Step 5: Find the combined similarity using Step 1 to Step 4

Step 6: If combined similarity > Similarity Threshold then the queries and URLs are clustered using Agglomerative Clustering Algorithm [10]

Step 7: The similar queries in the clusters are used for recommendation

For example, consider the query \( Q_1 = \text{‘grooming in harrisburg pa’} \) given by the user 227 on 2006-03-22 16:39:29. Consider another query \( Q_2 = \text{‘parking garage in harrisburg pa’} \) given by the same user on 2006-05-07 00:37:17. Here the queries are triggered at different time periods. The proposed work is time independent, so the time occurrences of the queries are not considered. Totally 9 terms are available in \( Q_1 \) and \( Q_2 \), out of this three terms are similar.

\[\text{Keyword Similarity} \left( Q_1, Q_2 \right) = \frac{3}{9} = 0.3333\]

Consider another query \( Q_1 = \text{‘www. herbchambers.com’} \) given by the user 144 on 2006-04-23 09:23:48. Consider another query \( Q_2 = \text{‘midwestcenter’} \) given by the user 644 on 2006-03-02 17:35:59. Here the queries are triggered at different time periods by different users.

\[\text{Keyword Similarity} \left( Q_1, Q_2 \right) = 0/2 = 0\]

When the query \( Q_1=\text{‘www. herbchambers.com’} \) is supplied, the user clicks 2 URLs and for the query \( Q_2=\text{‘midwestcenter’} \) only 1 URL is clicked. For both the queries only one URL is commonly selected.

\[\text{URL Similarity} \left( Q_1, Q_2 \right) = 1/3 = 0.33\]

When the query \( Q_1=\text{‘www. herbchambers.com’} \) was supplied, the user views the concepts ‘car, 4 wheeler, automobile’. The concept ‘car’ is concentrated by another user for the query \( Q_2=\text{‘midwestcenter’} \). For the query ‘www. herbchambers.com’, the user does not concentrates on ‘2 wheeler, rented taxi’. The concept ‘rented taxi’ is not concentrated by another user for the query ‘midwestcenter’.

\[\text{Concept Similarity} \left( Q_1, Q_2 \right) = 0.25 + 0.2 = 0.45\]

The query cluster generation is based on the combined similarity measure.
Fig. 4: Query and concepts with positive and negative preferences - Tripartite representation

Combined Similarity (www.herbchambers.com, midwestcenter) = Keyword Similarity + URL Similarity + Concepts Similarity = 0 + 0.33 + 0.45 = 0.78

If the Similarity threshold measure is 0.75, the combined similarity measure satisfies the threshold value and the queries www.herbchambers.com, midwestcenter are considered as similar and they are merged in a query cluster. Figure 4 shows the tripartite representation of query and concepts.

RESULTS

The algorithm is implemented in JDK 1.6.0_24. All the experiments are performed in Intel Core i3 processor 2.53 GHz with Windows 7 Home Premium (64-bit) and 4 GB RAM. A sample experimental data as enunciated in Table 3 is prepared for this research with a pursuit to conduct the evaluation of combined similarity based query recommendation. Table 6 list the similarity between the queries in terms of query keywords and clicked URL. Table 7 highlights the similarity of unique queries related to the positive and negative concept preferences. Figure 5 depicts the combined similarity which is calculated by using the formula stated in section 3.3.

The number of queries exhibiting similarity in the specified range of threshold values is specifically brought out in Figure 6.

Table 8 shows the average similarity of various attributes discussed previously.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Average Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Keyword</td>
<td>0.4</td>
</tr>
<tr>
<td>URL</td>
<td>0.33</td>
</tr>
<tr>
<td>Positive preferences</td>
<td>0.33</td>
</tr>
<tr>
<td>Negative preferences</td>
<td>0.261</td>
</tr>
<tr>
<td>Query and URL</td>
<td>0.366</td>
</tr>
<tr>
<td>Query,URL and Positive Preferences</td>
<td>0.433</td>
</tr>
<tr>
<td>Query,URL, Positive and Negative Preferences</td>
<td>0.629</td>
</tr>
</tbody>
</table>
Figure 7 shows the attributes and its average similarity computed for the log entries given in Table 3. In case of the query submission, ‘psychiatric disorders’ by the user 227, the research pursued extended its suggestions on the subsequent queries of similarity besides, the URLs based on the previous click through data which is shown in Table 9. Here minimum threshold >0 is considered.

In another case of query ‘www.herbcambers.com’ supplied by the user 144, the proposed work for combined similarity recommended the queries and URLs for the same user of another user 644. Table 10 clearly picturises the query and URL recommendations for the user 144.

Moreover, the clustering process is carried out by the precocious exploration on the similarities witnessed on various attributes as shown in Figure 8.
CONCLUSION

Query recommendation is an important technique which provides the future queries based on the history of past queries and click through data available in the query log. The proposed approach combines the similarity measure between the queries in terms of query keywords, URLs and concepts based on both positive and negative preferences. The navigation occurs at any time period and the technique is time independent. The Query cluster contains set of similar queries which are used to provide the query recommendation to the user to frame the queries in future. When the user decides and clicks the URL for the given query, if the rank for the clicked URL is high then the concept ontology is framed for that URL. The concept ontology will be considered to find the similarities in future.

REFERENCES

15. Jiawei Han and M. Kamber, “Data Mining: Concepts and Techniques,” In Morgan Kaufmann publishers.