Query Recommendation in Hidden Web Search Engine using Web Log Mining Techniques

Khushboo Gulati
Student
Department of CSE
Manav Rachna College of Engineering, Faridabad

Narender
Assistant Professor
Department of IT
Manav Rachna College of Engineering

ABSTRACT
Today a large amount of information on the Web is available only via search interfaces-the users are required to type in the set of keywords in search form in order to get the desired results from some websites. These websites are generally referred to as the Hidden web or Deep Web. Traditional search engines crawlers cannot index such pages because there are no static links to them. But with continuous advancement in the search engine technologies, most of the traditional search engines can now locate these deep web sources. In this paper, a new approach of query recommendation in hidden web search engine is introduced that would recommend queries to users on the basis of the user browsing behavior.

Keywords
Hidden Web, Hidden Web Crawler, Web Mining, Query Recommendation.

1. INTRODUCTION
The amount of the information on the Web is increasing day-by-day thereby increasing the size of the Web databases as well. Search engines are the best tool to retrieve the required information from this large amount information residing on the Web. The results of the hidden web sources are present in the searchable databases. These results are dynamic in nature i.e. the response is generated only when the user types in a particular keyword in the search interface of the web page. Even a slight change in the user query changes the response shown by the deep web sources. Since this information is dynamic in nature so it becomes difficult for the traditional crawlers to index such pages because there are no static links to them. This part of the WWW which cannot be easily crawled by the traditional search engine crawlers is termed as Hidden Web. The hidden database forms the most important part of the hidden web and according to the white paper by the Bright Planet [2], the number of web databases sites has reached to the range of 43,000 to 96,000 and it is increasing day-by-day. The 2001 study revealed that at that time the Deep Web was approximately 400-500 times the size of the Surface Web. Today's Internet is significantly bigger with an estimated 555 million domains, each containing thousands or millions of unique web pages. As the Web continues to grow, so too will the Deep Web and the value attained from Deep Web content [1].

Search Engines are the programs that search the documents on the WWW for a particular user query (a set of keywords) and return a corresponding list of results on the basis of the entered query. Beforehand search engines used to apply traditional information retrieval techniques that used keyword based similarity to identify the required documents. The search results of this approach were, however, of poor quality. Later, various ranking techniques were employed to provide the users with the desired results.

In context with the Hidden Web, search engines were not able to crawl the deep web sources but recent studies have shown that the most commonly used search engine, Google can now index Hidden Web data [3]. Although many advances have been made in the Web search engine technologies, yet there are many situations in which the user is presented with undesired results. So in order to enhance the search results provided to the user, query recommendation came into picture. This approach used to recommend queries on the basis of user behavior.

In this paper, a novel approach for query recommendation in the area of Hidden Web using the techniques of [4] has been proposed. The approach recommends the user with a set of most similar and popular user queries. This approach works by maintaining a user query log. Then the user is asked to enter a query on the basis of which the similarity is calculated between the entered query and the user query log. Next the clusters are formed on the basis of a particular threshold value and thus the query is finally recommended.

2. RELATED WORK
A lot of work has been done in the area of Hidden Web which is described as follows:-

Sriram Raghavan, Hector Garcia-Molina [6] proposed a task-specific, human assisted approach for crawling the hidden web and named it as HiWE (Hidden Web Exposer). HiWE processed and extracted useful information by using a technique called LITE (Layout-based Information Extraction Technique). The architecture proposed by them allowed the crawler to focus only on desired pages and the human assisted approach allowed for automatic form filling.

Luciano Barbosa and Juliana Freire [10] proposed a Form Focused Crawler (FFC)which selects the links that lead to documents of interest, while avoiding the links that lead to off-topic regions. The FFC consist of three different types of qualifier-First is the Form Classifier which distinguishes between the searchable form and non-searchable form. Next is the Link Classifier that is trained to identify the links that are likely to lead to pages that contain search forms. Third is the Page Classifier which is trained to classify the pages as belonging to topics in taxonomy.

Alexandrous Ntoulas, Petros Zerfos, Junghoo Cho [7] proposed the effective policies for generating the queries automatically for single attribute forms. For accessing the content a sequence of steps is followed: First the user issues a query through the search interface. Then a list of links that are most appropriate for the issued query, are returned. The user

then identifies the most relevant link and follows the actual website by clicking on it.

Neelam Duhan and A. K. Sharma [4] proposed a new technique based on the query log analysis for result optimization and query recommendation in normal search engines. Web Mining techniques were applied to get the desired results.

Here, in this paper the techniques proposed by Neelam Duhan and A.K. Sharma [4] are used and clubbed with the field of Hidden Web to recommend queries to users.

3. PROPOSED APPROACH

The proposed system dynamically recommends query to the user on the basis of the user query log maintained

3.1 Proposed Architecture

The architecture of the proposed system is shown in Figure-1, which consists of two main sections:-one the Hidden Web Crawler section and the other the Query Recommendation Section.

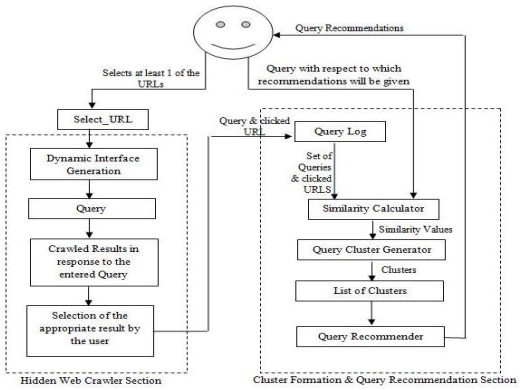

Figure-1: Architecture of the Proposed system

The system works in the following steps:

Initially the user is presented with three hidden websites, out of which the user selects at least one of the sites for interface generation. Then the procedure starts as follows-

Part-I: Deep Web Crawling

Step 1: After the selection of website user is presented with the corresponding search interface of the selected website in which the user can type in the query.

Step 2: Next, the hidden crawler collects the data from the selected sites on the basis of the user query entered and the result is displayed on the screen.

Step 3: From the displayed result, the user selects the most appropriate result. And this result gets stored in the User Query Log.

Part-II: Formation of Clusters

Step-1: For the formation of clusters, the user first enters the desired query.

Step-2: On the basis of the entered query, using the query similarity calculator, the similarity between the entered query and the each entry of User Query Log is calculated.

Step-3: Once the similarity is calculated, then on the basis of a threshold value, clusters are formed.

Step-4: The first cluster amongst the list of clusters so formed is the most favorable query for the user i.e. it forms the recommended query.

The significant components of the proposed architecture are explained in the next section.

3.2 Query Similarity Calculator

This approach functions on two principles: similarity based on the query keywords and cross-references. These principles are explained as follows:

3.2.1 Similarity Based on Query Keywords [4]

If two user queries have the same or similar terms, they usually point to same or similar information needs. The formula for calculating the content based similarity between two queries is given as below:

$$Sim_{keyword}(x, y) = \frac{|KW(x,y)|}{|kw(x) \cup kw(y)|} \quad \textbf{(1)}$$

where KW(x,y) represents the set of common keywords in the queries x and y, kw(x) and kw(y) are the sets of keywords in queries x and y respectively.

3.2.2 Similarity Based on User Feedback [4]

In this type of similarity calculation technique, two queries are regarded as similar queries if they both share or result in the selection of same or similar documents. Beeferman and Berger's Agglomerative clustering algorithm **[5]** forms the base of this principle. The approach is content ignorant, i.e. the algorithm functions without making the use of the actual content of the documents and queries in clustering. In the proposed system, the similarity analyzer calculates the similarity on the basis of the number of clicks made for a particular URL in response to the entered user query. This approach can be explained further with the help of a bipartite graph as shown in figure-2:

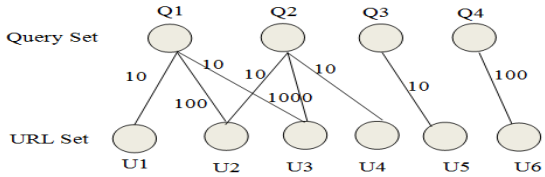

Figure-2: Bipartite Graph of Query Log

If two queries *x* and *y* share a common URL *u*, then similarity value is the ratio of the total number of distinct clicks on *u* with respect to both queries and the total number of distinct clicks on all the URLs accessed for both queries. If more than one URL is accessed then numerator is obtained by summing up the URL clicks of all common URLs. The formula that describes the similarity function on the basis of URL clicks is as follows:

$$Sim_{clickURL}(x,y) = \frac{\Sigma_{ui \in CU(x) \cap CU(y)}(LC(x,ui)+LC(y,ui))}{\Sigma_{zi \in CU(x) \cap CU(y)}(LC(x,zi)+LC(y,zi))} \quad \textbf{(2)}$$

where *LC(x,u)* and *LC(y,u)* represent the number of clicks on URL *u* corresponding to queries *x* and *y* respectively. *CU(x)* and *CU(y)* represent the sets of clicked URLs corresponding to queries *x* and *y* respectively.

The use of this formula can be explained with the help of an example. Consider 2 queries *Q1* and *Q2* which share

common URLs *U2* and *U3*, whereas URLs *U1*, *U4* are accessed by one of them (as shown in figure-2). The similarity between these queries can be calculated as:

$$Sim_{clickURL}(Q1,Q2)= \frac{(100+10)+(10+1000)}{(10+0)+(100+10)+(1000+10)+(10+0)} = 0.982$$

The calculated similarity values always lie between the range of 0 and 1. The formula given in (2) considers two queries similar by applying a threshold value on their similarity value.

3.2.3 Combined Similarity Measure

Both the approaches mentioned in section-3.2.1 and 3.2.2, have their own plus points. The former one helps in grouping together the queries of similar composition whereas the latter one takes the advantage of user's judgments. Thus both of them can partially capture the interest of user when considered individually. Hence, it is better to combine them in a single measure. The formula for this combination is given as follows:

$$Sim_{combined}(x,y)= \alpha \bullet Sim_{keyword}(x,y)+\beta \bullet Sim_{clickURL}(x,y) \quad (3)$$

where α and β are constants with $0 \le \alpha,\beta \le 1$ and $\alpha+\beta=1$. The value of these constants can be decided by the analysts depending on their needs and the importance being given to the two similarity measures. In the proposed system, these constants have been assigned a value of 0.5 each

3.3 Query Cluster Generator

Query clusters represent clearly defined user searching behavior while using search engines. The query cluster generating module retrieves these clusters by following the algorithm [4] as shown in figure-3. Each run of the algorithm creates k clusters. Since the user query log is dynamic in nature, therefore this query clustering algorithm should be incremental in nature.

The algorithm runs in a very simple manner: initially all queries are not assigned to any of the cluster. Each query is examined against all other queries (whether classified or unclassified) by using (3). If the similarity value results in a value which is above the pre-specified threshold value (τ), then the queries are grouped into the same cluster. The same process is repeated until all queries get classified to any one of the clusters.

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

Given: A set of *n* queries and corresponding clicked URLs stored in an array $Q[q_i,URL_1,..,URL_m],1 \le i \le n$
$\alpha=\beta=0.5$
Similarity threshold τ

Output: A set C={C1,C2,..,Ck} of *k* query clusters

//Start of Algorithm
k=1;
For (each query *p* in Q)
Set ClusterId(*x*) = Null; //initially no query is selected
For (each $p \in Q$)
{
ClusterId(x)=Ck;
Ck={x};
For (each $y \in Q$ such that $x \neq y$)
{

$$Sim_{keyword}(x, y) = \frac{|KW(x,y)|}{|kw(x) \cup kw(y)|}$$

$$Sim_{clickURL}(x,y) = \frac{\Sigma_{ui \in CU(x) \cap CU(y)}(LC(x,ui)+LC(y,ui))}{\Sigma_{zi \in CU(x) \cap CU(y)}(LC(x,zi)+LC(y,zi))}$$

$$Sim_{combined}(x,y)= \alpha \bullet Sim_{keyword}(x,y)+\beta \bullet Sim_{clickURL}(x,y)$$

If ($Sim_{combined}(x,y) \ge \tau$) then
Set clustered(y)=Ck;
Ck=Ck∪{y};
Else
continue;
}// end for
k=k+1;
}// end outer for
Return Query Cluster set C.

Figure-3: Algorithm for Clustering the Queries

3.4 Query Recommender

This component provides the user with a set of recommended queries. The recommended queries are the queries that are similar to the user submitted query and thus are contained in the cluster of that query. For example, the recommendations of a user query *Hidden Web* are:

Hidden Websites

Hidden Web

Hidden Web Crawler

Hidden Web Search Engine

When user submits a query, its keywords are matched with queries in the Query Cluster database and the most matched query is returned by the Query Recommender tool in a separate list box.

4. EXPERIMENTAL RESULTS

This section covers the practical results of the proposed system.

When the user first interacts with the system, he/she is presented with the page as shown in figure-4. The user selects the desired website and based in the user selection the corresponding search interfaces are presented to the user. The user can then type in his/her query. Figure-5 shows the results displayed for a user query "*Maruti Swift Mumbai*". From the displayed results table, the user can select the desired entry as per his suitability. For example if the first entry of the table (as shown in figure-5) is selected, then that query-URL pair i.e. *Maruti Swift Mumbai* and *autonagar.com* gets stored in the user query log.

For the query recommendation part, the user is allowed to enter a query for which he/she seeks recommendation. This is done with the help of the User query Log entries. Table-1 shows the replica of the actual User Query Log of the proposed architecture. This replica is used here for similarity calculations.

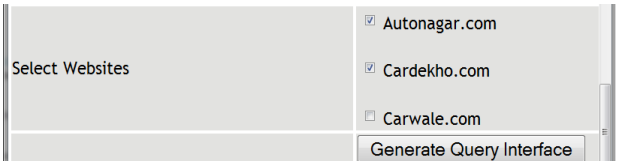

Figure-4: URL Selection

Click Me!	Make_Model	Years	Mileage	Price	Color	City	Listed_on	Url
Click Me !	Maruti Swift VXi BS IV	2006	68440	Rs. 2,60,000/-	Red	Mumbai	10/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift VXi BS IV	2006	68440	Rs. 2,60,000/-	Red	Mumbai	10/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift VXi BS IV	2006	35000	Rs. 2,75,000/-	Green	Mumbai	09/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift VXi BS IV	2006	53378	Rs. 2,45,000/-	Gray	Mumbai	09/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift LXi BS IV	2007	54000	Rs. 2,00,000/-	Red	Mumbai	09/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift VXi BS IV	2007	47000	Rs. 2,25,000/-	Gold	Mumbai	09/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift VXi BS IV	2008	25000	Rs. 2,50,000/-	Black	Mumbai	08/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift VXi BS IV	2009	22000	Rs. 3,50,000/-	White	Mumbai	07/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift VXi BS IV	2008	45000	Rs. 3,40,000/-	Brown	Mumbai	07/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift VXi BS IV	2006	46000	Rs. 2,28,500/-	Black	Mumbai	07/05/2014	http://www.autonagar.com
Click Me !	Maruti Swift VXI	2011 Model	20,383	Rs 4.85 lakh	Red	Mumbai	Last updated: 10 May	http://www.cardekho.com
Click Me !	Maruti Swift 2004-2010 Vdi BSIII	2010 Model	61,200	Rs 4.5 lakh	Black	Mumbai	Last updated: 10 May	http://www.cardekho.com

Figure-5: Results Displayed

The user for example types in the query as *Maruti Swift Faridabad* and *cardekho.com.* Now on the basis of this query the similarity values are calculated as follows:

Similarity on the basis of the Query Keywords

x=Maruti Swift Faridabad and *y=Maruti Swift Delhi* (first entry of Table-1)

$$Sim_{keyword}(x, y) = 2/6 = 0.333$$

Similarity on the basis of URL

x=Maruti Swift Faridabad, cardekho.com and *y=Maruti Swift Delhi, cardekho.com*

$$Sim_{clickURL}(x,y) = (3)+(2+9) / (3)+(2+9+12)$$

$$= 14/26 = 0.538$$

Combined Similarity

$$Sim_{combined}(x,y) = (0.5)(2/6) + (0.5)(14/26) = 0.4355$$

Similarly, this calculation will be performed with each individual entry of the User Query Log and stored in a separate table. Then on the basis of a specific threshold value, cluster generation will take place using algorithm shown in figure-3. The first cluster so formed will become the recommended query for the user.

Table-1.Sample Query Log for Calculations

S No.	Query	Clicked URL	Clicks
1.	Maruti Swift Delhi	www.cardekho.com	2
2.	Maruti Swift Faridabad	www.cardekho.com	3
3.	Chevrolet Beat Mumbai	www.autonagar.com	6
4.	Hyundai Santro Delhi	www.cardekho.com	5
5.	Chevrolet Beat Faridabad	www.cardekho.com	10
6.	Hyundai Santro Noida	www.carwale.com	5
7.	Maruti WagonR Chandigarh	www.autonagar.com	7
8	Chevrolet Spark Gurgaon	www.carwale.com	12
9.	Maruti Swift Delhi	www.cardekho.com	9
10.	Chevrolet Beat Delhi	www.autonagar.com	10
11.	Maruti Swift Delhi	www.carwale.com	12

5. CONCLUSION AND FUTURE WORK

A new approach of query recommendation for deep web sources is proposed. This would simplify the searching experience of user because the recommended queries depend upon the user's feedback and browsing behavior. More the number of queries fired; more will be the range of recommended queries.

Currently this system works with 3 websites but in future if the numbers of websites in this search engine are increased then it would prove to be more beneficial for the users as they can now find the data of different websites at a single location. Moreover, advanced web mining techniques can be used to further improve the functioning of this system.

6. REFERENCES

[1] BrightPlanet.com, "Understanding the Deep Web in 10 Minutes", http://brightplanet.com

[2] BrightPlanet.com, "The Deep Web: Surfacing Hidden Value", http://brightplanet.com.

[3] Jayant Madhavan, David Ko, Lucja Kot, "Google's deep-web crawl", VLDB '08, August 24-30, 2008, Auckland, New Zealand, p-1241-1252.

[4] Neelam Duhan, A.K. Sharma, "Rank optimization and query recommendation in search engines using web log mining techniques", Journal of Computing, Volume 2, Issue 12, December 2010, ISSN 2151-961, p-97-104.

[5] Doug Beeferman and Adam Berger, 2000,"Agglomerative clustering of a search engine query log". In proceedings of the 6th ACMSIGKDD International Conference on Knowledge Discovery and Data Mining, (August). ACM Press, New York, NY, 407-416.

[6] Sriram Raghavan, Hector Garcia-Molina, "Crawling the hidden web", in Proceedings of the 27th VLDB Conference, 2001.

[7] Alexandrous Ntoulas, Petros Zerfos, Junghoo Cho, "Downloading textual hidden web content through keyword queries", in P*roc*.5th ACM/IEEE, Joint Conference on Digital Libraries(JCDL), 2005, p 100-109

[8] Tantan Liu, Gagan aggarwal, "Stratification based heirarchical clustering over a deep web data source", pg-70-81.

[9] Supriya, Meenakshi Sharma, "Deep web data mining", dInternational Journal of IT, Engineering and Applied Sciences Research (IJIEASR),Volume 2, No.3, March 2013, p 69-71.

[10] Luciano Barbosa, Juliana Freire, "Searching for hidden-web databases", in *Proc* the 8th International Workshop on the Web and Databases (webDB 2005), June 16-17, 2005.

[11] 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.