Learning Based Approach for Search Engine Selection in Metasearch

R.Kumar¹, A.K Giri²
¹Ph.DScholar, School of Computer and System Sciences Jawaharlal Nehru University New Delhi, INDIA
²Ph.DScholar, School of Computer and System Sciences Jawaharlal Nehru University New Delhi, INDIA

ABSTRACT

In recent years, the web has become a huge source of information, which is mostly unstructured in the form of text or image. However, every search engine uses its own method or algorithm for ranking the retrieved results. The main goal of Metasearch over the single search engine is increased coverage and a consistent interface to ensure that result from several places can be meaningfully combined. In this paper we propose a learning based Query similarity using rank merge list of document approach for search engine selection algorithms to identify the most useful search engines that are likely to contain the relevant documents for the user query. The objective of search engine selection is to improve efficiency as it would result in sending a query to potentially useful underlying search engines only. Finally, it concludes the paper by pointing out some open issues and possible direction of future research related to search engine selection.

Keyword- Query, search engine, meta-search engine, query similarity, MRDD algorithms.

I. INTRODUCTION

In recent years, the web has become a huge source of information, which is mostly unstructured in the form of text or image. The people all over the world poses queries using their favorite search engine to find relevant information. However, every search engine uses their own method for ranking the retrieved results. The Metasearch engines usually send the query simultaneously to different search engines resulting in queries being processed in parallel thereby saving time. Metasearch engine [2] is a tool that allows searching multiple search engines at the same time and returning more comprehensive and relevant document that satisfy the information needs of the user, efficiently. In other words, Metasearch engine is a system that provides unified access to multiple existing search engines [3]. When user poses a query to the Metasearch through the user interface, the Metasearch engine is responsible to identify appropriate underlying search engine which has relevant document with respect to the user query. Each search engine has a text (unstructured data) database, defined by the set of documents that can be searched by search engine. All underlying search engines retrieve most of the relevant documents, which the Metasearch engine combines into single ranked list and displays them to the user. A ranking of the document is based on the user query, top rank document has high query weight. The main goal of Metasearch over the single search engine is increased coverage and a consistent interface [1]. A consistent interface is necessary for the Metasearch engine to ensure that result from several places can be meaningfully combined while the user is not aware about the underlying search engine. Two types of the search engines exist[3] namely general purpose search engine and special purpose search engine. The general purpose search engine aims to provide the capability to search all types of the web page with respect to the user query like Google, Altavista, Excite, Lycos and HotBot etc [3]. The special purpose search engine retrieves the document for a defined domain such as specific subject area. Example, Cora search engine focuses on computer science search paper and Medical World Search focus to retrieve the medical information. It is believed that hundreds of thousands of special purpose search engines currently exist on the web [4]. The motivation for the Metasearch engine [5] includes, (1) An increase in the search coverage at the rate at which the web has been increasing is much faster than the indexing capability of a single search engine. (2) Metasearch engine effectively combines the coverage of all underlying search engines. (3) Retrieves relevant document by merging documents retrieved from the underlying search engine and ranking them with respect to the user query thereby making it convenient and reliable for the user to retrieve relevant information. (4) Facilitate the invocation of multiple search engines.

The user in order to retrieve most relevant document needs to first identify search engines with the most relevant information followed by sending queries to each search engine in the corresponding query format.

The rest of the paper is organized as follows: Section II, describes about the Metasearch engine, its component, In section III related work for the search engine selection approach. The proposed Query Similarity with rank list document using a Round Robin algorithm is presented in...
section IV. In section V shows the algorithms based example. The experimental simulation is shown in section VI. Finally, a conclusion and future work discuss in section VII.

II. (A) METASEARCH ENGINE

Metasearch engine is a tool that sends the user requests to multiple primary search engines combine the results retrieved from them together and display them to the user. The information on the web has increased rapidly over time. When a query is submitted to the Metasearch engine, there is some question arise in the mind. (1) Which underlying search engine will be selected for a user query. (2) How to preprocess the submitted query with respect to better utilization of the underlying search engine and (3) How to merge result retrieved from the search engines. All decisions are taken by the Metasearch engine based on keyword of user queries. The challenging aim of Metasearch engine is the selection of appropriate search engines and merging results retrieved from them, with respect to the user query.

II (B). ARCHITECTURE OF METASEARCH ENGINE

Information on the digital library is stored in disparate sources and each of these sources has their own search capability. In general, any organizations have their own web site and also have their own search engine. When user poses a query to find relevant information, the Metasearch engine finds such information using its components. The main components of a Metasearch engine are search engine selector, document selector, query dispatcher, and result merger. The aim of Metasearch is to maximize the precision or retrieval effectiveness while minimizing the cost. In order to carry out this, the meta search engines select the most appropriate search engine containing relevant information. Each selected search engine should retrieve as much relevant information as possible. The architecture of a Metasearch engine, as in [3], is shown in Figure (1) The Metasearch engine consists of four components namely Search Engine Selector, Document Selector, Query Dispatcher, and Result Merger.

Search Engine Selector: The search engine selector selects the appropriate underlying search engine with respect to the user query. A good search engine selector should correctly identify search engines while minimizing identifying irrelevant search engines. The approaches for selecting search engines are discussed later in this paper.

Document Selector: The document selector determines what documents to retrieve from the selected search engines. The aim is to retrieve more relevant documents with few irrelevant documents. To find out the relevant information different similarity measure is used which estimate the relevance between document and user query. The similarity is measured based on a pre-defined threshold value. The high similarity value shows that the information is more relevant with respect to the user query.

Query Dispatcher: The query dispatcher has a mechanism to establish a connection of a server with each selected search engine in order to dispatch query to each of these search engines. In general, the user query will be sent to the search engine after preprocessing. Every search engine may or may not have the same query as posed on the Metasearch engine.

Result Merger: The result merger merge document retrieved from the selected search engines. The result merger combines all the result into a single ranked list and arranges the documents in descending order with their global similarity with respect to the user query. The top most documents having higher global similarity in the ranked list are returned to the user through the interface.

III. RELATED WORK

A good search engine selector uses the different selection algorithms to identify potentially useful search engine for a given user query. Search engine selection main task is to identify the most useful search engines that are likely to contain relevant documents for the user query. The objective of search engine selection is to improve efficiency as it would result in sending a query to only potentially useful underlying search engines. In paper [7], utilizes the retrieved results of past queries for selecting the appropriate search engines for a specific user query. The selection of the search engines is based on the value of relevance between user query and the search. Modeling Relevant Document Distribution (MRDD) [8] [3] is a static learning based approach, which uses a set of training queries for learning. With the help of training queries, it identifies all the relevant documents returned from every search engine and arrives at a distribution vector for each relevant document. Similarly, it finds the distribution vector for each training query and calculate the average distribution vector is used to identify the appropriate search engines. In ref. [9], each component search engine is represented by a set of pairs of the document frequency of the term and the sum of the weights of the term over all documents in search engines. A threshold is associated with each query in [9] to indicate that only documents
IV. PROPOSE ALGORITHM

The propose an algorithm inspired by [7] algorithm in which the retrieved documents for each past query from all selected search engines, used to calculate the relevance between search engines and respective past query using top k document. The top k document is a rank merge list of documents from all search engines but in the top k documents may have some common documents which are not avoid to making the rank merge list of the documents. In proposing algorithms, build a rank merge list using Round Robin algorithms [18] which is avoiding the common documents in the rank merge list. Utilizes the retrieved results of past queries for selecting the appropriate search engines for a specific user query. In algorithm rel\(s_j|q\), means it is more appropriate search engine contain more relevant information for the user query. The value of rel\(s_j|q\) depends on rel\(s_j|p_i\) and \(\text{sim}(p_i|q)\), where rel\(s_j|p_i\) is the relevance between search engines and past queries and \(\text{sim}(p_i|q)\) is the similarity between all past queries with the user query. The search engines with higher value of rel\(s_j|q\) being selected by the Metasearch engine.

The selection of the search engines is based on the value of relevance between user query and the search engine say rel\(s_j|q\), which indicate the how likely it is for the search engine to be relevant to the user query \(q\) or what percentage of the relevant documents for the user query \(q\) is in the search engine \(s_j\). Ranking in the search engines is carried out according to the value of rel\(s_j|q\). A higher value of Rel\(s_j|q\) implied that the search engine contains most relevant documents with respect to user query \(q\). Thus, the search engines having higher value for Rel\(s_j|q\) is selected.

Algorithm

Input: Let set \(H[p, q, s]\), where \(p\) is the number of past queries, \(q\) is the user query, and \(s\) is the number of search engine.

Output: Ranked list of topmost search engines.

Method:

Step1: For each \(i^{th}\) past query
Rank the documents retrieved from all search engines into the single ranked list

Step2: Compute

\[
\text{Rel}\left(\frac{s_j}{p_i}\right) = \frac{\sum_{\text{top} T \text{ doc in the merged ranked list}} \text{Rel}(s_j/doc)}{T}
\]
Where T is a pre-defined number of top documents.

If \( \text{doc} \in s_j \) then \( \text{Rel}(s_j/\text{doc}) = 1 \)

Otherwise \( \text{Rel}(s_j/\text{doc}) = 0 \)

**Step3:** Generate the ranked list \( R_s \) of documents returned from the search engines for the user query \( q \)

**Step 4:** For each \( i^{th} \) past query \( p_i \) and user query \( q \), compute the similarity between past query and user query using 
\[
\text{Sim}(p_i/q) = \frac{1}{\|R_{p_i}\|} \sum \text{Score}(\text{doc}, R_{p_i} \cap R_q)
\]

where \( R_{p_i} \) and \( R_q \) are ranked list of past query and user query returned from the search engines and Score function is calculated as

\[
\text{Score}(\text{doc}, R_{p_i} \cap R_q) = 1 - \frac{\text{doc ranked in } R_{p_i}}{\|R_{p_i}\|} - \frac{\text{doc ranked in } R_q}{\|R_q\|}
\]

**Step5:** Normalized the value of \( \text{Sim}(p_i/q) \) using

MaxSim\( _q = \max \text{Sim}(p_i/q) \)

CutSim\( _q = 0.8 \times \text{MaxSim}_q \)

Normalized

\[
\text{Sim}(p_i/q) = \begin{cases} 
0 & \text{if } \text{Sim}(p_i/q) < \text{CutSim}_q \\
\frac{\text{Sim}(p_i/q) - \text{CutSim}_q}{\text{MaxSim}_q - \text{CutSim}_q}, & \text{otherwise}
\end{cases}
\]

**Step 6:** For user query \( q \)

For (each \( j^{th} \) search engine) Compute

\( \text{Rel}(s_j/q) = \sum \text{Rel}(s_j/p_i) \times \text{Sim}(p_i/q) \)

**Step 7:** Ranked the search engine according to the value of \( \text{Rel}(s_j/q) \). A larger value of \( \text{Rel}(s_j/q) \) is more likely means it contains most relevant documents with respect to the user query \( q \).

### V. ALGORITHM BASED EXAMPLE

Let the set of past queries be \( TQ = \{Tq_1, Tq_2, Tq_3\} \) and the set of search engines be \( SE= \{Se_1, Se_2, Se_3, Se_5, Se_6\} \) and a user query be \( q \) are input to the algorithm.

**Step1:** Let assume all past queries having maximum three terms.

\[
\begin{array}{ccc}
Tq_1 & 1 & 3 & 2 \\
Tq_2 & 3 & 4 & 2 \\
Tq_3 & 5 & 2 & 1 \\
\end{array}
\]

For every past query \( Tq_i \), Metasearch engine selects the underlying search engine with respect to the training queries are shown as

\[
\begin{array}{cccccccc}
Tq_1 & Se_1 & Se_3 & Se_4 & Se_5 & Se_6 \\
Tq_2 & Se_2 & Se_4 & Se_5 & Se_6 \\
Tq_3 & Se_1 & Se_2 & Se_3 & Se_5 & Se_6 \\
\end{array}
\]

Now apply the all training query to all selected search engines and assume the following document are retrieving with respect to the user query.

\[
\begin{array}{cccccccc}
Tq_1 & Se_1 & Se_3 & Se_4 & Se_5 & Se_6 \\
Tq_2 & Se_2 & Se_4 & Se_5 & Se_6 \\
Tq_3 & Se_1 & Se_2 & Se_3 & Se_5 & Se_6 \\
\end{array}
\]

**Step 2:** Computes \( \text{Rel}(Se_j/Tq_j) \) between a search engine and all training queries for top-8 documents

For \( Tq_1 = \{1,3,2\} \)

\[
\begin{array}{cccccccc}
\text{Rel}(Se_1/Tq_1) & \text{Rel}(Se_1/Tq_1) & \text{Rel}(Se_1/Tq_1) & \text{Rel}(Se_1/Tq_1) \\
0.500 & 0.333 & 0.416 & 0.333 \\
\end{array}
\]

For \( Tq_2 = \{3,4,2\} \)

\[
\begin{array}{cccccccc}
\text{Rel}(Se_1/Tq_2) & \text{Rel}(Se_1/Tq_2) & \text{Rel}(Se_1/Tq_2) & \text{Rel}(Se_1/Tq_2) \\
0.500 & 0.583 & 0.416 & 0.500 \\
\end{array}
\]

For \( Tq_3 = \{5,2,1\} \)
Step 3. Now apply the user query \( q = \{5, 3, 2\} \) to all selected search engines by Metasearch.

<table>
<thead>
<tr>
<th>( Se_i )</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>D5</th>
<th>D6</th>
<th>D7</th>
<th>D8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Se1</td>
<td>6</td>
<td>7</td>
<td>15</td>
<td>8</td>
<td>12</td>
<td>3</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>Se2</td>
<td>2</td>
<td>5</td>
<td>11</td>
<td>9</td>
<td>8</td>
<td>11</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>Se5</td>
<td>7</td>
<td>13</td>
<td>12</td>
<td>13</td>
<td>12</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Se6</td>
<td>2</td>
<td>15</td>
<td>10</td>
<td>13</td>
<td>11</td>
<td>9</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

Single merge list of documents return by the six search engines for the user query UQ using Round Robin algorithms [18]

\[
doc_i \begin{array}{cccccccc}
6 & 2 & 7 & 5 & 13 & 15 & 12 & 10 & 8 & \ldots & T
\end{array}
\]

Step 4. Now find the similarity between user query and all past queries

| \( Sim(Tq_i|q) \) | \( Sim(Tq_2|q) \) | \( Sim(Tq_3|q) \) |
|------------------|------------------|------------------|
| 0.5556           | 0.4514           | 0.6111           |

Step 5. Normalized the value of \( Sim(Tq_i|q) \)

\[
\begin{array}{ccc}
Normalized \ Sim(PQ_i|q) & 0.5455 & \\
Normalized \ Sim(PQ_2|q) & 0 & \\
Normalized \ Sim(PQ_3|q) & 1.000 & \\
\end{array}
\]

Step 6. Find the value of \( Rel(s_j/q) \) are shown as below.

<table>
<thead>
<tr>
<th>( Rel(Se_1/q) )</th>
<th>0.0000</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Rel(Se_2/q) )</td>
<td>0.5715</td>
</tr>
<tr>
<td>( Rel(Se_3/q) )</td>
<td>0.1325</td>
</tr>
<tr>
<td>( Rel(Se_4/q) )</td>
<td>0.6061</td>
</tr>
<tr>
<td>( Rel(Se_5/q) )</td>
<td>0.4000</td>
</tr>
<tr>
<td>( Rel(Se_6/q) )</td>
<td>0.1716</td>
</tr>
</tbody>
</table>

Step 7. The search engines are ranked according to the value of \( Rel(s_j/q) \). The higher value of \( Rel(s_j/q) \) is the most appropriate for the user query. The order of selected search engine for user query are \( \{Se_4, Se_2, Se_5, Se_6, Se_3, Se_1\} \).

VI. EXPERIMENTAL SIMULATION

In order to testify the effectiveness of the propose algorithm is simulated in MATLAB 2010b. In this experiment we assume that there are 30 search engines, six training query and a user query. The simulation result of a \( q_Sim \) algorithm using rank merge list of document and [3] algorithms. In Figure (3) is showing the higher value of relevance between a search engine and user query and in figure (4) the average distribution vector (ADV) for all search engines are showing that is the most appropriate search engine with respect to the user query. In algorithm [3], ADV is showing the average documents retrieved from all search engines which is relevant to the user query, and that is not effective and robustness compare to the proposed algorithms.

Figure (3)

![Figure (3)](image)

Figure (4)

![Figure (4)](image)
It is observed that the two algorithms tend to select a reasonably high number of search engines in common when a reasonable high number of search engines are selected by them.

V. CONCLUSION

This paper proposes query similarity using rank merge list of documents that is the effectiveness and robustness compare to the algorithm [3]. The proposed algorithms are retrieved to relevant document compare to the algorithm [3] because of pruning the common documents and provide the rank document list to compute the relevance between a search engine and user query. The purpose of retrieved the relevant document is to find the appropriate search engine for user query. The proposed algorithms, the relevance between a search engine and user query is find out with the help of the rank merge list that gives the refined and the relevant search engine because of voiding common documents in the merge list.

Future Work

The problem of search engine selection would be compounded if large numbers of search engines are selected for the user query. As a result, it would be more difficult to choose the relevant search engine among them. Therefore, it is a challenging task to select the relevant search engine among the various search engines with respect to the user queries. Using the revolutionary algorithms like genetic algorithm, particle swarm optimization (PSO), magnetic optimization algorithms, simulated annealing and ant colony optimization algorithms we can optimize the search engine selection which is the most relevant to the user queries.

REFERENCES


