Asian Journal of Information Technology 5 (11): 1256-1261, 2006

Query Fine Tuning and Search Results Reranking Using Content Measure and Context Reference Algorithm

[1]Angelina Geetha, [2]R. Srinivasan and [3]A. Kannan
[1]Department of Information Technology, [2]Department of Computer Science,
B.S.A. Crescent Engineering College, Anna University, India
[3]Department of Computer Science and Engineering, Guindy College of Engineering,
Anna University, India

Abstract: In this study, we propose a method to improve the precision of top N retrieved documents retrieved from the web by re-ordering the retrieved documents from a search engine. The user query is accepted and the search process is initiated by employing an external search engine. On the retrieved search results, content analysis is carried out and various measures of relevance are calculated. Based on the overall relevance measure, the search results are reranked. The search context plays a vital role in framing of the query and search process. Hence we propose an algorithm to perform the context analysis on the reranked results. The benefit of this is two fold. First, the user is given a preview about on what context the keywords are used in a document thus reducing the irrelevant document browsing time. Second, by viewing the context, the user can fine tune the search query to get a closer search result. From the experimental results we have found that the reranking based on our relevance measure shows improvement in the search result obtained from search results.

Key words: Searching, reranking, filters, query weighing, term frequency, context search, content matching, context analyzer

INTRODUCTION

Due to the rapid growth of World Wide Web, the task of the search engines has become very challenging. The task of a typical search engine is to accept a query (set of keywords), search the Internet and give the sites whose contents are most relevant to the user query. The search results are ranked by the search engine based on some pre determined algorithms. By experiments, it is found that these ranking of search results are not very efficient. This is because the documents retrieved by the web crawlers are very large in number and often documents that are retrieved are not related to user query. This led to the need for reranking algorithms.

A reranking algorithm accepts the search results from a search engine, reranks the results based on various measures. Filters like link filter and content filters are widely used for this purpose. In our reserch, we combine the content filter and context filter for implementing the reranking algorithm. Relevancy ranking is the method that is used to order the result list in such a way that the web documents that are more relevant to the user query will be placed at the beginning.

Moreover, the user frames a query, the context of the search is very important. Many a times the context in which the user frames a query is not explicit. Hence there is a need to filter the results based on the context of search also. Our system accepts the user query, accepts the search results from any external search engine. On the obtained search results content analysis and context analysis is carried out. Based on the content analysis, the documents are reranked. Based on the context analysis, the user is provided with a user interface screen by which the user can browse through the context of the keywords without actually down-loading those web documents. If the user is interested in the content of the document, then he can download and view the full document.

Corresponding Author: Angelina Geetha, Department of Information Technology, B.S.A. Crescent Engineering College, Anna University, India

Related work: Ranking search results is a fundamental problem in information retrieval. Most common approaches primarily focus on similarity of query and a page, as well as the overall page quality[1-3].

The objective of a ranking function is to match the documents in a text collection against a query and order them in descending order of their predicted relevance. In the vector space model, both documents and queries are represented by vectors. Suppose there are a total of t index terms in the whole collection, a given document D and query Q can be represented as follows:

$$D = (wd1, wd2, wd3,..., wdt)$$
$$Q = (wq1, wq2, wq3,..., wqt)$$

where wdi, wqi (i = 1 to t) are term weights assigned to different terms for the document and query respectively.

The similarity between a query and a document can be calculated by the widely used cosine measure given by Salton[4]. Documents are then ordered by decreasing values of this measure. In the vector space model, these weights are commonly measured by their statistical properties or statistical features. For example, one of the most widely used statistical features in term weighting strategy is Term Frequency (TF), which measures how many times the term has appeared in the document or query.

Many methods have been proposed to rerank documents. Lee *et al.* proposed a document reranking method based on document clusters. They build a hierarchical cluster structure for the whole document set and use the structure to rerank the documents[5]. Balinski *et al.*[6] proposed a document reranking method that uses the distance between documents to modify initial relevance weignts. Luk *et al.*[7] use the title information of documents to rerank documents. Crouch *et al.*[8] use the unstemmed words in the queries to reorder the documents. Xu *et al.*[9] make use of global and local information to do local context analysis and then use the information acquired to rerank documents. Qu *et al.*[10] use manually built thesaurus to rerank retrieved documents. Each term in a query topic is expanded with a group of terms in the thesaurus. Bear *et al.*[11] use manually crafted grammars for topics to reorder documents by matching grammar rules in some segments of an article. Kamps[12] proposes a reranking method based on assigned, controlled vocabularies. Yang *et al.*[13] use query terms that occur in both query and top N (N<=30) retrieved documents to rerank documents.

In our research we propose a document reranking algorithm where the user selects a document as the seed for the reranking procedure. The similarity weightage is calculated based on query key term weightage, document term frequency and document distance. A dynamic distance weight table is generated. The shortest path algorithm decides the reordering of the documents.

Our algorithm initially accepts a query from the user. Extracts the key terms from the query based on extraction rules. The top N (N<= 50) search results are acquired from any search engine. Our focus is more on the content analysis which gives an insight into the frequency of keywords, their distribution and the context of appearance in the web document. The content of these web documents are analyzed and the measures of relevance to the given query is calculated. The search results are reranked based on the relevancy score. Moreover the context in which the keywords appear in the document are extracted and presented to the user using an interface. Based on the context, the user is allowed to fine tune or reframe the query to enhance the search process.

SYSTEM ARCHITECTURE

The system architecture is given in Fig. 1. The overall system architecture is given in Fig. 1. To initialize the search process, the user gives a query. This query is fed to the search system. The stop words are removed from this query and the key terms are given to any external search engine to search the Internet.

The top N results (N = 50) are retrieved and given as input to the content analyzer. At this stage, the contents of the web documents are analyzed and various measures are calculated. Based on these measures, the web documents are reranked and passed on to the context analyzer. The context analysis algorithm makes use of the content analysis results and generates an interface for the user to go through the context of keywords and key phrases in the reranked web documents. After browsing the context of usage of the keywords and key phrases,

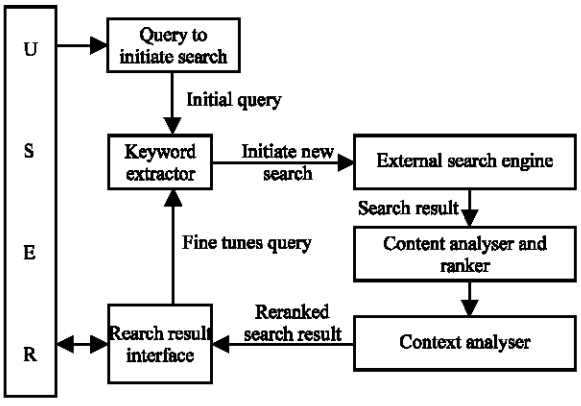

Fig. 1 : System architecture

Fig. 2 : Content analysis and reranked results

the user can view the entire content of the document by clicking on it. The user can also fine tune the query, by reframing the query which goes through the entire process again.

Query and search process: When the query is received from the user, the stop words are removed from the query. For this purpose and external dictionary is integrated. The extracted keywords converted into a string by placing '+' symbol between them and this is given to a search engine (say Google) and the search is triggered. The results from the search engine are captured and the system stores the URLs of the search result documents in a database for reranking by the content analyzer

Content analyzer and reranker: In this phase, the URLs of the search result documents are retrieved from the database. Every web document is retrieved and detagged. The system has been developed in Java and we have restricted our work to only web documents of HTML format and text format. The task of feature extraction focuses on the key term extraction. All the stop words are removed. Stemming of words are also considered. For example 'network', 'networking', networked' are considered alike. The following measures are calculated. The term frequencies of the key terms are tabulated. Term Frequency (TF) is how many times a particular key term has occurred in the document.

The favorable keywords are the keywords that occurred in the search query. The term frequency of favorable keywords is denoted by $FK_{freq.}$ The term frequency of all keywords is denoted by $K_{freq.}$ The percentage of distribution of favorable keywords over the overall frequency of keywords in the document is calculated as P1.

$$P1 = FK_{freq}/K_{freq} * 100$$

In our research we have not only considered the key terms, but also the keyphrases. Considering the time delay we have restricted our key phrases to a length of two words only. If there are n key terms we generated a Key Phrase Matrix (KPM) of size n x n and compute the frequency of the occurrence of the key phrases.

$$KPM(i,j) = x ,$$

indicates that key terms i and j occur next to each other x times. We have considered KPM(i,j) as equal to KPM(j,i). For example the phrase 'Programming in network' is considered as the same key phrase as 'network and programming'. Hence the upper diagonal matrix alone has to be calculated. KPM(i,i) is ignored. Though the key terms are very high, we found that the KPM is highly sparse and does not need very high memory storage, since we considered storing only the nonzero elements of the matrix.

The favorable key phrases are the key phrases which contain the keywords that occurred in the search query. The term frequency of favorable keyword phrases is denoted by FKP_{freq}. The term frequency of all keyphrases is denoted by KP_{freq}. The percentage of distribution of favorable keywords over the overall frequency of keywords in the document is calculated as P2.

$$P2 = FKP_{freq} / K_{freq} * 100$$

The size of the web document also plays a vital role in calculating the relevance frequency. The ratio of the frequency of the favorable keywords and key phrases and the document size (in KBytes) is given by P3.

$$P3 = (FK_{freq} + FKP_{freq}) / Doc_{size} * 100$$

P3 gives the indication about the distribution of favorable keywords and keyphrases in the document.

Using the measures P1, P2 and P3 the documents are reranked. The following Fig. 2 gives the result screen of the search results before and after reranking. The 'Expand result' button is used to initiate the context analysis algorithm as explained below.

Fig. 3 : Context analyzer interface

CONTEXT ANALYSIS ALGORITHM

The context analysis algorithm is designed to extract the context of keywords and key phrases in the reranked web documents. The natural language sentence formation rules are adopted for extracting the context of the given keyword..

Algorithm:

For every web document d of the reranked search result

{

For every keyword Ki in d, where i is the ith keyword in the document

Scan the document and extract the context of occurrence based on sentence analysis rules

For every key phrase KPi in d

Scan the document and extract the context of occurrence based on sentence analysis rules

Eliminate duplicates if any

Display the keyword and key phrases along with their context of occurrence in tree structure

}

The Fig. 3 shows the sample interface for the query data structures and algorithms. Initially there are two stubs keyword and key phrases. When the user clicks on keywords, the list of keywords in that document extracted by the algorithm is displayed. By clicking of a specific keyword, its context of occurrence is exploded. The same is extended for key phrases also. A scroll option provides easy visibility. If the user wishes to reframe the query, based on a different context, he can go back to the search screen Fig. 2 and reframe his search query. Thus the user is allowed to fine tune his query until the exact context of his search is established. Once the user is satisfied with the context of reference of his search, the whole document can be viewed by clicking on the document title. This saves the time of browsing the irrelevant web documents.

RESULTS AND DISCUSSION

Basic assumption for the analysis was that, the top N search results are considered closer to the search query. We have limited our experimental analysis to top 10 ranked documents given by any search engine (say Google). For a sample search of keywords "Data Structures+Algorithms", the various measures of reranking are computed given in Table 1.

Table 1: Measures of reranking

Measures of Relevance	Page link									
	1	2	3	4	5	6	7	8	9	10
P1	5.20	8.38	4.53	14.42	14.02	9.26	50.00	25.58	7.09	15.63
P2	9.41	16.86	7.80	33.33	33.33	15.79	50.00	31.25	20.00	35.71
P3	1.67	2.50	1.76	4.80	4.00	2.67	4.00	4.00	2.64	5.00

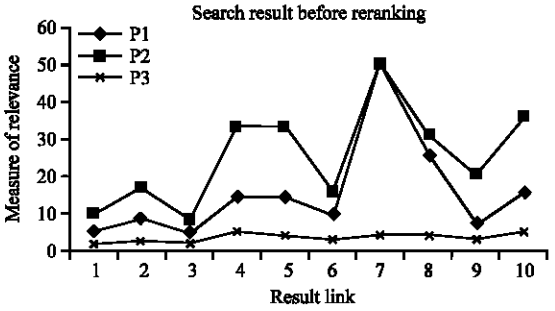

Fig. 4 : Search results before reranking

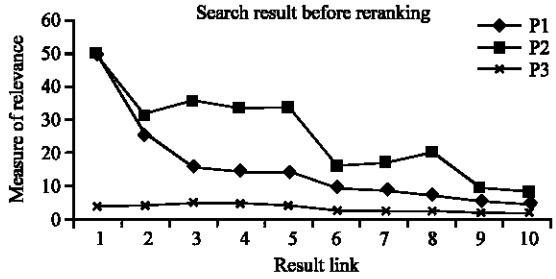

Fig. 5 : Search results after reranking

The graph shown in the Fig. 4, displays the measure of relevance as per the ranking given by the search engine.

The graph shown in Fig. 5 gives the measure of relevance after reranking by our algorithm. It is clearly shown by these two graphs that by employing the reranking algorithm, there is significant change in the measure of relevance even among the top ten results given by the search engine.

CONCLUSION

Search engines are enhancing their search algorithms so that it is possible to answer the user queries effectively. Various ranking and reranking algorithms have been developed in the past to cater to the basic need of the Internet user, which is searching for the required information. Our algorithm employs the content analysis to rerank the documents based on the relevance score of the web document with the given search query. Moreover, the context analyzer extracts the context in which the keyword is handled in the documents, which helps the user to redefine or modify the existing query

for more focused search. Our future work is focused on upgrading the reranking algorithm with semantic analysis feature to enhance the search process. This algorithm can be extended by including the other relevancy scores such as position score, query term weightage.

REFERENCES

1. Baeza-Yates R. and B. Ribeiro-Neto, 1999. Modern Information Retrieval, Addison-Wesley.
2. Brin, S. and L. Page, 1997. The Anatomy of a Large-scale Hypertextual Web Search Engine, In Proceedings of WWW Conference.
3. Salton, G. and M. McGill, 1983. Introduction to Modern Information retrieval, McGraw-Hill.
4. Salton, G. and C. Buckley, 1988. Term-weighting approaches in automatic text retrieval, Information Processing and Management, 24:513-523.
5. Lee, K., Y. Park and K.S. Choi, 2001. Document reranking model using clusters, Information Processing and Management, 37: 1-14.
6. Balinski, J. and C. Danilowicz, 2005. Reranking method based on inter document distance, Information Processing and Management, 41:759-775.
7. Luk, R.W.P. and K.F. Wong, 2004. Pseudo relevance Feed Back and Title Reranking for Chinese IR, In Proceedings of the NTCIR Workshop 4.
8. Crouch, C., D. Crouch, Q. Chen and S. Holtz, 2002. Improving the retrieval effectiveness of very short queries, Information Processing and Management.
9. Xu, J. and W.B. Croft, 2000. Improving the effectiveness of information retrieval with local context analysis, ACM Transactions on Information Systems, 18: 79-112.
10. Qu, Y.L., G.W. Xu and J. Wang, 2000. Rerank Method Based on Individual Thesaurus, Proceedings of the NTCIR2 Workshop.
11. Bear, J., D. Israel, J. Petit and D. Martin, 1997. Using Information Extraction to Improve Document Retrieval, Proceedings of the Sixth Text Retrieval Conference.
12. Kamps, J., 2004. Improving Retrieval Effectiveness by Reranking Documents Based on Controlled Vocabulary", Proceedings of the 21st European Conference on Information Retrieval.
13. Yang, L.P., D.H. Ji, G.D. Zhou and Y. Nie, 2005. Improving Retrieval Effectiveness by Using Key Terms in Top Retrieved Documents, Proceedings of the 27th European Conference on Information Retrieval.