An Efficient Cluster Based Searching Process for Finding Keyword Query Related Documents

Putchakayala Mahesh Reddy [1], Mula Sudhakar [2]
Final M.Tech Student[1], Asst.professor[2]
[1,2]Dept of CSE, *Sarada Institute of Science, Technology and Management (SISTAM), Srikakulam, Andhra Pradesh*

Abstract

Now a day's to analyse an efficient query related document has not been work the difficulties of queries over database. So many researches are proposed many methods for predicting query related text documents. By implementing these techniques are not given an efficient keyword query related documents. By overcome those types of problems we are implementing a keyword query related interface is used to assign each query term to schema element in the database. So that the test result type must be desired and also get query related text documents. Some of the existing methods are not empirical to show direct adaptation of ineffective for structured data. By overcome those problems in this paper we are proposed an efficient keyword query related process for getting efficient search result. By implementing efficient keyword query related process we can perform the best search process on a text documents. In the efficient keyword query related process mainly contains four concepts i.e. text pre-processing, build mvs matrix, clustering of text document and performing searching process. By implementing those concepts we can get the efficient query related text document.

Keywords

Searching techniques, Prediction, Keyword, query, K Means Clustering Algorithm, Distance, Frequency, Local Frequency, global Frequency.

I. INTRODUCTION

As the amount of electronic data continues to grow, the availability of effective information retrieval systems is essential. Despite a continuing increase in the average performance of information retrieval systems, the ability of search systems to find useful answers for individual queries still shows a great deal of variation [1]. An analysis of the chief causes of failure of current information retrieval (IR) systems concluded that, if a search system could identify in advance the problem associated with a particular search request, then the selective application of different retrieval technologies should be able to improve results for the majority of problem searches [2]. The ability to predict the performance of a query in advance would enable search systems to respond more intelligently to user requests. For example, if a user query is predicted to perform poorly, the user could be asked to supply additional information to improve the current search request. Alternatively, a search system could selectively apply different techniques in response to difficult and easy queries, for example the selective application of different retrieval models, or automatic relevance feedback. Query performance prediction is the problem of trying to identify, without user intervention, whether a search request is likely to return a useful set of answers. The importance of the query difficulty prediction problem has been highlighted in the IR community in recent years; the Text REtrieval Conference (TREC) Robust tracks in 2004 and 2005 included an explicit query difficulty prediction task [1], and prediction has been the 3ubject of specific workshops [4]. Despite this recent growth in attention, the prediction of query difficulty is an open research problem.

We present several predictors of query performance. The predictors are concerned with pre-retrieval prediction. The information required by such prediction is obtained from various collection, document and term occurrence statistics. These are all obtained at indexing time, and can be efficiently fetched from inverted index structures that are widely used in information retrieval [4]. The computation of these predictors can therefore be carried out prior to query evaluation. This has significant advantages in terms of simplicity and efficiency, factors whose importance increases as the size of collections continues to grow. We propose two broad classes of pre-retrieval predictors: first, predictors that are based on the similarity between queries and the collection; and second, predictors that are based on the variability of how query terms are distributed in the collection, by exploring the in-document statistics for the input queries.

II. RELATED WORK

Many different approaches for the prediction of query performance have been proposed. These can be divided into three broad categories: pre-retrieval predictors, post-retrieval predictors, and learning predictors. In this paper we focus on pre-retrieval predictors; the background section therefore concentrates on previous work in this area. We also provide brief descriptions of the other families of predictors for completeness. Pre-retrieval predictors can be calculated from features of the query or collection, without requiring the search system to evaluate the query itself. The information that these predictors use is available at indexing-time; they are therefore efficient, and impose a minimal overhead on the retrieval system. Pre-retrieval predictors generally make use of evidence based on term distribution statistics such as the inverse document frequency, inverse collection term frequency, or the length of a query. A range of pre-retrieval predictors were proposed and evaluated by He and Ounis [5]. Their experimental results showed the two best-performing predictors to be the average inverse collection term frequency (AvICTF), and the simplified clarity score (SCS). In their approach, the SCS is obtained by calculating the Kullback-Leibler divergence between a query model and a collection model. We use AvICTF and SCS as baselines in our experiments, and these approaches are explained in detail in Section 4. Scholer et al. [6] describe results based on using the inverse document frequency (IDF) to predict query performance. They find that using the maximum IDF of any term in a query gives the best correlation on the TREC web data. We present results using the maximum IDF (MaxIDF) as a baseline in our experiments. Post-retrieval predictors use evidence that is obtained from the actual evaluation of the underlying search query. These predictors can leverage information about the cohesiveness of search results, and can therefore show high levels of effectiveness. However, for the same reason they are less efficient: the search system must first process the query and generate an answer set, and the answer set itself is then usually the subject of further analysis, which may involve fetching and processing individual documents. This can impose a substantial overhead on a retrieval system. Cronen-Townsend et al. [7] proposed a post-retrieval predictor based on language models: they calculate the divergence between a statistical model of the language used in the overall collection and a model of the language used in the query, to obtain an estimate of the ambiguity of the query. Unlike the simplified clarity score pre-retrieval predictor discussed previously, this approach estimates the query language model from the documents that are returned in the answer set of a retrieval system. The approach was demonstrated to be highly effective on newswire data. Post-retrieval predictors for web data were developed by Zhou and Croft [8], who use a weighted information gain approach that shows a high correlation with system performance for both navigational and informational web search tasks. Other post-retrieval predictors have considered factors such as the variability of similarity scores; for example, Kwok et al. divide a search results list into groups of adjacent documents and compare the similarity among these [9]. Zhou and Croft [10] introduced ranking robustness scores to predict query performance, by proposing noise channel from information theory. This approach has shown higher effectiveness than the clarity score.

III.PROPOSED SYSTEM

The main objective of proposed system is to perform the efficient query search and reduce the time complexity of in the searching process. In this paper we are proposed an efficient query searching process i.e. topic based cluster search algorithm. By implementing this algorithm we can get efficient search result and also reduce time for searching the query. Before performing the search the query we can take sample document and search query in that documents. The implementation procedure of topic based cluster algorithm is as follows.

A) Text Pre-processing:

In the text pre-processing we can get only text formatted data for searching query. Before performing search operations we can get all documents and reduce all tag in that document. After getting each document text we can find out relative frequency (R_{freq}) of each document. Before finding relative frequency we also find local and global frequency of each word in the document. The local frequency (L_{freq})of each can be calculated by number of occurrence of each word in the document. After finding local frequency of each word in the document we can find out global frequency(G_{freq}) . Using both frequencies we can find out relative frequency of each document by using following formula.

$$R_{freq} = L_{freq} + G_{freq} / 2.0$$

After finding relative frequency we can calculate document weight of each document by using following formula.

N=size of each document

$L_{freq=}$ Local frequency of each word in the document

G_{freq} =Global Freqency of each document

$$\text{Weight (W)} = L_{freq} * \text{Math.Log}(N/G_{freq}) + 0.01$$

By using that formula we can calculate each document weight. After we can create MVS Matrix of each document to other documents.

B) *Build MVS Matrix*:

In the generation of MVS matrix we can calculate cosine similarity each document to other document. Based on MVS matrix we can perform the clusterization of documents. The cosine similarity of any two document can be find by using following equation.

d1= Total number of words in first document
d2= total number of words in second document
d_{prd}= d1*d2
$d1_{sqr}$ = d1*d1
$d2_{sqr}$ d2*d2

d_{sqrprd} =$d1_{sqr}$ * $d2_{sqr}$

sim=d_{prd}/d_{sqrprd}

By using those formulas we find out each document cosine similarity and also we generate matrix formatted data. likewise we can calculate cosine similarity of each document to other document and arranged in the form matrix.

k means clustering algorithm for grouping related documents:

By calculating of MVS matrix we can perform the clusterization process. By performing clusterization process we can grouping all relating document into single group. Before performing clusterization we get all cosine similarity of each document to other document. Based on cosine similarity of each document we can perform clusterization process. The step of clusterization process is as follows.

1. Enter the number of cluster for performing clustering of document.
2. After that finding number of documents are available in the database.
3. Randomly choose the centroid of document based on number of clusters we want.
4. After finding centroid document we can get cosine similarity of each centroid document.
5. After that we can also get remaining document of cosine similarity.
6. Find out distance of each centroid to other document based on cosine similarity by using following formula

```
for (int i=0;i<docs.size();i++)
    {

      int minInd =0;
      double mindis=0;

      for (int j=0;j<k;j++)
      {
       double dis = cosSim(docs.get(i),getCentriod(clusturs[j]));
        if(j==0 || mindis>dis)
        {
         minInd=j;
         mindis=dis;
        }
      }

      clusturs[minInd].add(docs.get(i));

    }.
```

By using that code we can find out related documents in a group. After grouping all related document into group perform the searching operation in those groups and get only query matched document.

C) Topic based searching process:

In this module we perform the searching operation of query in the document. In this we can get each cluster document and convert into text format. After that we can search each word in that cluster and find out the word id existing in that group or not. So that if the word is existing in the that cluster we display that document in the cluster. Likewise we can search all cluster document and get only the query related cluster document. By implementing those concepts we can get more effective search result and also time complexity for performing search operation.

IV. CONCLUSIONS

In this paper we are proposed a novel problem for performing effective searching operation in documents. By implementing this concept we can improve more efficiency of searching operation and also reduce time complexity. In this paper we are proposed topic based cluster searching algorithm for finding related document of query search. In this algorithm we can find out each document cosine similarity and also find out distance of centroid document to other documents. After finding distance we can perform clusterization process by using k means clustering algorithm. After performing clustering process we can perform the searching process for query. By performing query search we can get all query related documents of clusters can be display. By implementing those concepts we can improve efficiency in the searching operation.

REFERENCES

[1] Voorhees, E.M.: Overview of the TREC, robust retrieval track. In: The Fourteenth Text REtrieval Conference (TREC 2005), Gaithersburg, MD, 2006. National Institute of Standards and Technology Special Publication 500-266 (2005).

[2] Harman, D., Buckley, C.: The NRRC reliable information access (RIA) workshop. In: Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval, Sheffield, United Kingdom, pp. 528–529 (2004).

[3] Carmel, D., Yom-Tov, E., Soboroff, I.: SIGIR workshop report: predicting query difficulty - methods and applications. SIGIR Forum 39(2), 25–28 (2005)

[4] Zobel, J., Moffat, A.: Inverted files for text search engines. ACM Computing Surveys 38(2) (2006).

[5] He, B., Ounis, I.: Query performance prediction. Information System 31(7), 585–594 (2006).

[6] Scholer, F., Williams, H.E., Turpin, A.: Query association surrogates for web search. Journal of the American Society for Information Science and Technology 55(7), 637–650 (2004).

[7] Cronen-Townsend, S., Zhou, Y., Croft, W.B.: Predicting query performance. In: Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval, Tampere, Finland, pp. 299–306 (2005).

[8] Zhou, Y., Croft, W.B.: Query performance prediction in web search environments. In: Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, pp. 543–550 (2007).

[9] Kwok, K.L.: An attempt to identify weakest and strongest queries. In: Predicting Query Difficulty, SIGIR 2005 Workshop (2005).

[10] Zhou, Y., Croft, W.B.: Ranking robustness: a novel framework to predict query performance. In: Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval, Arlington, Virginia, pp. 567–574 (2006)