Review: Predicting The Efficiency of Difficult Keyword Queries Over Databases

Miss. Varsha Vetal, Mrs. Sanchika Bajpai.

Abstract:

Data mining is the process of discover patterns in large data sets. In data mining it extract the information from large data set and transferred to the another structure which can be understandable to user. Now days keyword search is used by many organizations. In relational database the keyword search used to find tuples by keyword queries. But this method lies in low performance so we must find the keyword queries over databases to increase the query performance. Keyword queries over databases gives easy access to data or information, but it having the problem of low ranking quality. So it is useful to identify queries which having low ranking quality problem for improving the satisfaction as well as performance of difficult query. The framework measures the degree of difficulty for a hard keyword queries over databases. For this the basic terms are used like cluster analysis, anomaly detection, dependencies. Methods uses for performance of information retrieval system are measure in relevant document and non-relevant document like precision and recall which discover problem of low ranking. As well as keyword query interface used to provide flexibility and ease of use in searching data.

Index Terms: Data mining, Databases, query performance, **keyword query.**

I. INTRODUCTION:

The data mining task is the automatic or semi-automatic analysis .It is used for large quantities of data to extract previously unknown interesting patterns. These are patterns are classified into three groups :

- Cluster analysis:
  Cluster analysis includes the groups of data records of patterns that is it grouping a set of objects. These objects are lies in same group called cluster. Cluster analysis is not algorithm but it gives solution for algorithm. Such algorithm that is clustering algorithms are based on cluster model.
- Anomaly detection:
  Anomaly detection is also known as outlier detection .It includes unusual records in data mining. It gives identification of data items.
- Dependencies:
  Dependencies include the association rule mining. It discovers relations between variables in large databases.

  Keyword query interfaces (KQIs) for databases provides flexibility and ease of use in searching and exploring data.[1][5].keyword queries have potential answer in data set. Keyword query interfaces identifies the information needs or requirements behind every keyword query. It rank the answer so it appears at the top of the list.[1][6].

Database is the collection of data and this data is organizes to model used to support internal operations. In this a computer program quickly selects pieces of data. Databases contain entities, and entities contain attributes that take attribute

values.Some of the hardness of answering a query are as follows: First, unlike queries in languages like SQL, users do not normally specify the desired schema element(s) for each query term. For instance, query Q1 Godfather on the IMDB database (http://www .imdb .com) does not specify if the user is interested in movies whose title is Godfather or movies distributed by the Godfather company. Thus, a KQI must find the desired attributes associated with each term in the query. Second, the schema of the output is not specified, i.e., users do not give enough information to single out exactly their desired entities .For example; Q1 may return movies or actors or producers.

Keyword query interface (KQIs) evaluated on well-known IMDB data set. This data set contains structured information about movies and people in given business. IMDB database mainly contain three tables that is actor , director and movies.

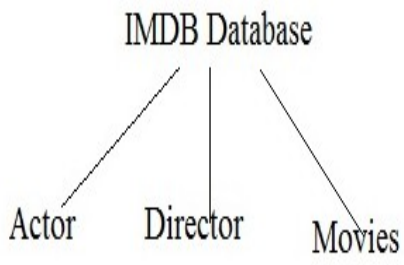


Fig.1.IMDB database Table

Methods uses for performance of information retrieval system are measure in relevant document and non-relevant document .These are having terms as follows:

- Precision:
  It is the fraction of documents retrieved that are relevant to user's information need.

- Recall:
  It is the fraction of the documents that are relevant to the query that are successfully retrieved to user.
- Average Precision:
  Precision and Recall are single value metrics that belongs to whole document list which are returned by system. Computation of Precision and recall ranks the sequence of document. Average precision computes the average value of documents.
- Mean Average Precision (MAP):
  MAP is used for set of of queries is the mean of the average precision scores for every query.

$$MAP = \frac{\sum_{q=0}^{Q} AveP(q)}{Q}$$

Where Q is number of queries.

II. LITERATURE SURVEY:

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things are satisfied, ten next steps are to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system.

In this paper, we analyze the features of difficult queries over databases. It propose novel method to detect such queries. We take advantage of the structure of the data to gain insight about the degree of the difficulty of a query given the database. We have implemented some of the most popular and representative algorithms for keyword search on databases such as SR algorithm and used them to evaluate our techniques

.The results show that our method predicts the degree of the difficulty of a query efficiently and effectively. [4]

Pros:

1. Keyword search provides user friendly interface rather than Xpath and Xquery.
2. XML is used to store data in XML document format rather than table format.
3. XML provide security to data, user not easily recognize the XML data rather than traditional Table format.
4. User enters a keyword (i.e. Attribute, key, identifier).
5. SR scores measures the difficulty of queries over database.
6. The algorithm to compute the SR score, and parameters to tune its performance.
7. It gives small time overhead compared to the query execution time.

III. METHODOLOGY:

1.Methods:

Researchers have proposed methods to predict hard queries over unstructured text documents. We can broadly categorize these methods into two groups:

1) pre-retrieval
2) post-retrieval

1) Pre-retrieval method :

This method predict the difficulty of a query without computing its results. These methods usually use the statistical properties of the terms in the query to measure *specificity*, *ambiguity*, or *term-relatedness* of the query to predict its difficulty.

2) Post-retrieval methods:

This method utilize the results of a query to predict its difficulty and generally fall into one of the following categories.

- *Clarity-score-based*: The methods based on the concept of *clarity score* assume that users are interested in a very few topics

- *Ranking-score-based:* The ranking score of a document returned by the retrieval systems for an input query may estimate the similarity of the query and the document.[5]

- *Robustness-based*: Another group of post-retrieval methods argue that the results of an easy query are relatively stable against the perturbation of queries, documents or ranking algorithms.[6].

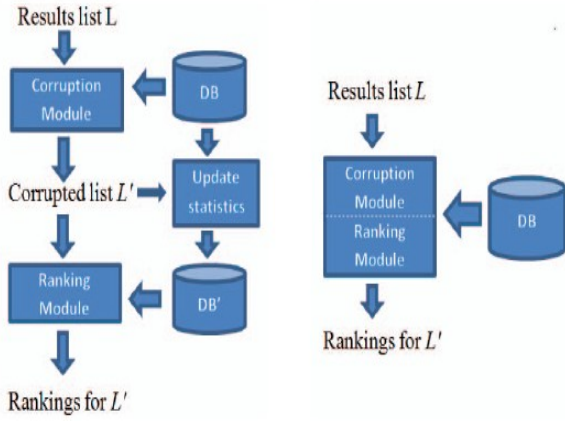
2. Algorithm:-

Structured Robustness Algorithm

Algorithm shows the Structured Robustness Algorithm (SR Algorithm), which computes the exact SR score based on the top *K* result entities. Each ranking algorithm uses some statistics about query terms or attributes values over the whole content of DB. Some examples of such statistics are the number of occurrences of a query term in all attributes values of the DB or total number of attribute values in each attribute and entity set. These global statistics are stored in *M* (metadata) and *I* (inverted indexes) in the SR Algorithm pseudocode. SR Algorithm generates the noise in the DB on-the-fly during query processing. Since it corrupts only the top *K* entities, which are anyways returned by the ranking module, it does not perform any extra I/O access to the DB, except to lookup some statistics. Moreover, it uses the information which is already computed and stored in inverted indexes and does not require any extra index.

Fig. 2.(a) shows the execution flow of SR Algorithm. Once we get the ranked list of top K entities for Q, the corruption module produces corrupted entities and updates the global statistics of DB. Then, SR Algorithm passes the corrupted results and updated global statistics to the ranking module to compute the corrupted ranking list. SR Algorithm spends a large portion of the robustness calculation time on the loop that re-ranks the corrupted results (Line 13 in SR Algorithm), by taking into account the updated global statistics. Since the value of K (e.g., 10 or 20) is much smaller than the number of entities in the DB, the

top K entities constitute a very small portion of the DB.



(a) (b)

Fig.2.Execution flow of SR algorithm And SGS – Approx (a)SR algorithm (b)SGS –Approx.

Thus, the global statistics largely remain unchanged or change very little. Hence, we use the global statistics of the original version of the DB to re-rank the corrupted entities. If we refrain from updating the global statistics, we can combine the corruption and ranking module together. This way re-ranking is done on-the-fly during corruption. SGS-Approx algorithm is illustrated in Fig. 2.(b).

CONCLUSION

We introduced SR algorithm for difficult keyword queries over databases .The problem of predicting the effectiveness of difficult keyword queries over databases is introduced in this paper. We showed that the current prediction methods for queries over unstructured data sources cannot be effectively used to solve this problem. We set forth a principled framework and proposed novel algorithms to measure the degree of the difficulty of a query over a DB, using the ranking robustness principle. Based on our framework, we propose novel algorithms that efficiently predict the effectiveness of a keyword query.

REFERANCES

1. V..Hristidis, L. Gravano, and Y. Papakonstantinou, "Efficient IRstyle keyword search over relational databases," in *Proc. 29th VLDB Conf.*, Berlin, Germany, 2003, pp. 850–861
2. Y. Luo, X. Lin, W. Wang, and X. Zhou, "SPARK: Top-k keyword query in relational databases," in *Proc. 2007 ACM SIGMOD*, Beijing, China, pp. 115–126.
3. V. Ganti, Y. He, and D. Xin, "Keyword++: A framework to improve keyword search over entity databases," in *Proc. VLDB Endowment*, Singapore, Sept. 2010, vol. 3, no. 1–2, pp. 711–722.
4. J. Kim, X. Xue, and B. Croft, "A probabilistic retrieval model for semistructured data," in *Proc. ECIR*, Tolouse, France, 2009, pp. 228
5. A. Nandi and H. V. Jagadish, "Assisted querying using instant-response interfaces," in Proc. SIGMOD 07 , Beijing, China,pp. 1156–1158.
6. O. Kurland, A. Shtok, D. Carmel, and S. Hummel, "A Unified framework for post-retrieval query-performance prediction," in Proc. 3rd Int. ICTIR, Bertinoro, Italy, 2011, pp. 15–26.
7. S. Cheng, A. Termehchy, and V. Hristidis, "Predicting the effec-tiveness of keyword queries on databases," in Proc. 21st ACMInt. CIKM , Maui, HI, 2012, pp. 1213-1222.
8. C. Hauff, L. Azzopardi, D. Hiemstra, and F. Jong, "Query perfor-mance prediction: Evaluation contrasted with effectiveness," in Proc. 32nd ECIR, Milton Keynes, U.K., 2010, pp. 204–216.

