A Survey on Novel Method for Prediction of Difficult Keyword Queries

Sonal G. Chopde & Prof. S. A. Murab
Jagdambha College of Engineering and Technology, Yavatmal, Maharashtra 445105

Abstract: Keyword queries are used to access data from databases. To improve the performance of querying system it would be useful to identify queries with low ranking quality. In this paper we analyze the characteristics of hard queries and propose a novel framework to measure the difficulty for a keyword query over a database. We evaluate our query difficulty prediction model against two effectiveness benchmarks for popular keyword search ranking methods. The ranking qualities of the results provide a good user satisfaction. Our intensive experiments show that the algorithms predict the issue of a question with comparatively low errors and negligible time overhead.

1. Introduction

Keyword query interfaces for databases have attracted a lot of attention within the last decade because of their flexibility and easy use in looking and exploring the data. [1][5]. since any entity in a data set that contains the query keywords is a potential answer, keyword queries typically have much possible answer. Keyword queries should determine the information desires behind keyword queries and rank the answer so the required answers seem at the highest of the list [1] [6].Some of the difficulties of answering a query 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 are interested in movies whose title is Godfather or movies distributed by the Godfather Company. Therefore, 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 [7]. For example, Q1 may return movies or actors or procedures. There are cooperative efforts to produce standard benchmarks and evaluation platforms for keyword search methods over databases. One effort is the data-centric track of INEX Workshop [8]. Wherever KQIs square measure evaluated over the well known IMDB information set that contain structured information regarding movies and other people in show business. Queries were provided by participants of the workshop. Another effort is the series of Semantic Search Challenges at Semantic Search Workshop [9], where set is that the Billion Triple Challenge data set at http://vmlion25.deri.de. It’s extracted from completely different structured data sources over the online like Wikipedia. The queries are taken from Yahoo! Keyword query log. It is necessary for a KQI to acknowledge such queries and warn the user or use various techniques like question reformulation or question suggestion. It’s going to additionally use techniques like question results diversification. To the most effective of our data, there has not been any work on predicting or analyzing the difficulties of the query over databases. Researchers have proposed some methods to detect difficult queries over plain text document collections [10][13]. However, these techniques aren’t applicable to our drawback since they ignore the structure of the information. Above all, as mentioned earlier, a KQI should assign every question term to a schema element(s) within the information. It should additionally distinguish the specified result type(s). We tend to through empirical observation show that directs diversifications of these techniques area unit ineffective for structured data.

2. Literature Review

Researchers has been propose a methods to predict hard queries over unstructured text documents [10][13][17].We can broadly categorize these methods into two groups: pre-retrieval and post-retrieval methods. Pre-retrieval methods [14][18] predict the difficulty of a query that it does not utilize its result. These methods usually use the statistical properties of the term in the query to evaluate specificity, ambiguity, or term-relatedness of the query to predict its difficulty [19]. There are some examples of these statistical characteristics are average inverse document frequency of the query terms or number of documents that contain at least one query term [14]. We developed a method for predicting query performance by computing the relative entropy between a query language model and the corresponding collection language model. Post-
retrieval methods utilize the results of a query to predict its difficulty and generally fall into one of the following categories.

### 2.1 Clarity-score-based

It is based on the concept of clarity score assume that users are concerned in a very few topics. Thus, sufficiently noticeable from other documents in the collection [10][14][15]. It is efficient than pre-retrieval based methods for text documents.[10] Some systems compute the distinguish ability of the queries results from the documents in the collection by comparing the probability distribution of terms in the results with the probability distribution of terms in the whole collection. If these probability distributions are relatively similar, the query results contain information about almost as many topics as the whole collection, thus, the query is considered becomes the difficult [10]. Too many successors has been proposed methods to improve the efficiency and effectiveness of clarity score [14], [15]. However, one requires domain knowledge about the data sets to extend idea of clarity score for queries over databases. Each topic in a database contains the entities that contain a similar subject. It is therefore hard to define a formula that partitions entities into topics as it requires finding an effective similarity function between entities. Such similarity function depends mainly on the domain knowledge and understanding users’ preferences [21]. For instance, distinct attributes may have distinct impacts on the degree of the similarity between entities.

### 2.2 Ranking-score-based

Ranking score based methods defines the ranking score of the document returned by retrieval system for a user query. It estimates the similarity between a query and the document and defines the difficulty of a query by the difference between weighted entropy score of top ranked result and the score of other documents,[16], [17]. Zhou and Croft shows that the information gained from a desired list of documents should be much more than the information gained from typical documents in the collection for an easy query. They measure the degree of the difficulty of a query by computing the difference between the weighted entropy of the top ranked results’ scores and the weighted entropy of other documents’ scores in the collection [17]. Shitok et al. argue that whatever the amount of non-query-related information in the top ranked results must be negatively correlated with the deviation of their retrieval scores [16]. Using language modeling techniques, they show that the standard deviation of ranking scores of top-k results calculate the quality of the top ranked results effectively. We examine the query difficulty prediction accuracy of this set of methods on databases and show that our model outperforms these methods over databases.

### 2.3 Robustness-based

Robustness-based methods defines how robust is the query over specific document. This method defines the degree of difficulty of query by considering the robustness of the ranking over two versions of data, original version and corrupted version, and compares the top k results of same query over these two versions.[12] Degree of difference between these results defines the degree of hardness of query. Some methods use machine learning techniques to study complex queries and its properties and then predict their hardness [22]. These methods are effective, if large amount and quality of data are available which are normally not available for many databases.

### 3. Basic Estimation Techniques:

#### Data sets:

The INEX data set is from the INEX 2010 Data Centric Track [8]. The INEX data set contains two entity sets: movie and person. Each entity in the movie entity set represents one movie with attributes like title, keywords, and year. The person entity set contains attributes like name, nickname, and biography. The SemSearch data set is a subset of the data set used in Semantic Search 2010 challenge [9]. The original data set contains 116 files with about one billion RDF triplets. Since the size of this data set is extremely large, it takes a very long time to index and run queries over this data set. Hence, we have used a subset of the original data set in our experiments. We first removed duplicate RDF triplets. Then, for each file in SemSearch data set, we calculated the total number of distinct query terms in SemSearch query workload in the file. We selected the 20, out of the 116, files that contain the largest number of query keywords for our experiments. We have converted each distinct RDF subject in this data set to an entity whose identifier is the subject identifier. The RDF properties are mapped to attributes in our model. The values of RDF properties that end with substring —#type” indicates the type of a subject. Hence, we set the entity set of each entity to the concatenation of the values of RDF properties of its RDF subject that end with substring —#type”. If the subject of an entity does not have any property that ends with substring —#type”, we set its entity set to —Undefined Type". We have added the values of other RDF properties for the subject as attributes of its entity. We stored the information about each entity in a separate XML file. We have removed the
relevance judgment information for the subjects that do not reside in these 20 files. The sizes of the two data sets are quite close; however, SemSearch is more heterogeneous than INEX as it contains a larger number of attributes and entity sets.

**Query Workloads:**

Since we use a subset of the dataset from SemSearch, some queries in its query workload may not contain enough candidate answers. We picked the 55 queries from the 92 in the query workload that have at least 50 candidate answers in our dataset. Because the number of entries for each query in the relevance judgment file has also been reduced, we discarded another two queries (Q6 and Q92) without any relevant answers in our dataset, according to the relevance judgment file. Hence, our experiments are done using 53 queries (2, 4, 5, 11-12, 14-17, 19-29, 31, 33-34, 37-39, 41-42, 45, 47, 49, 52-54, 56-58, 60, 65, 68, 71, 73-74, 76, 78, 80-83, 88-91) from the SemSearch query workload. 26 query topics are provided with relevance judgments in the INEX 2010 Data Centric Track. Some query topics contain characters “+” and “−” to indicate the conjunctive and exclusive conditions. In our experiments, we do not use these conditions and remove the keywords after character “−”. Some searching systems use these operators to improve search quality.

**Top-K results:**

Generally, the basic information units in structured data sets, attribute values, are much shorter than text documents. Thus, a structured data set contains a larger number of information units than an unstructured data set of the same size. For instance, each XML document in the INEX data centric collection constitutes hundreds of elements with textual contents. Hence, computing Equation 3 for a large DB is so inefficient as to be impractical. Hence, similar to [13], we corrupt only the top-K entity results of the original data set. We re-rank these results and shift them up to be the top-K answers for the corrupted versions of DB. In addition to the time savings, our empirical results in Section 8.2 show that relatively small values for K predict the difficulty of queries better than large values. For instance, we found that K = 20 delivers the best performance prediction quality in our datasets.

**Number of corruption iterations (N):**

Computing the expectation in Equation 3 for all possible values of x is very inefficient. Hence, we estimate the expectation using N > 0 samples over M (|A| × V). That is, we use N corrupted copies of the data. Obviously, smaller N is preferred for the sake of efficiency. However, if we choose very small values for N the corruption model becomes unstable.

**4. Conclusion**

We introduced the novel problem of predicting the effectiveness of keyword queries over DBs. We showed that 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 Method to measure the degree of the difficulty of a query over a DB, using the ranking robustness principle. Our extensive experiments show that the algorithms predict the difficulty of a query with relatively low errors and negligible time overheads.

**5. References**