Context Similar Diversification of Keyword Search Results using Different Approaches

Mrudul Arkadi, Prof. P.B.Mali
Department of Computer Engineering
Smt. Kashibai Navale college of Engineering
Pune, Maharashtra, India

Abstract

To gain knowledge one of the easiest and nearest solution nowadays is search your problem on internet. User needs solution in just no time. Basic solution to this is searching process. Searching can be done through keyword hunt. If user fires a query using multiple keywords or sentence it becomes quiet easy to get result which is contextually similar to fired query. But when search query is a single word, system has more chances of getting ambivalent or vague search results while considering context similarity scenario. Paper considers single keyword search and results into accurate outcome of related candidate queries. For that paper suggests four different approaches which refer to left and right context which leads to calculate occurrence of keywords, namely baseline solution, anchor pruning approach, naïve based mapping approach and advanced filtering approach. System moderately adds complexity to dataset and concludes for better results like time, accuracy and efficiency.

Keywords: Mutual information Score, SLCA, Anchor, Levenshtein, Baseline solution, weka, naïve based mapping, advanced filtering ,context-similarity, usefulness.

I. INTRODUCTION

Context provides extra information to help improve search result relevance. [9] Searching can be defined as the process of finding the documents that matches the user query. Searching can be done either by simple search and context based search. In simple search, all the documents are taken into account where as in context based search, only documents with relevant context are searched. This context based search have goal that is to place all the related keywords under one domain which influence the meaning of search query by using some methodology.

If users of information retrieval system are not professionals [7], if they do not know choosing the right words that best express their information needs, then context similarity based search is needed. This is motivated by the fact that the initial query returns a result that rarely meets the user's need, by providing more related words as a search result tries to reach till user's intension.

Paper uses DBLP database. When user searches a query, system needs to calculate all the context related words by using different approaches like baseline [10], anchor pruning [10], naïve based mapping and advanced filtering. Evaluation is done by increasing database complexity and system time required to complete the processing. Usefulness is dependent on user's intension. At last we show all the candidate queries we got from all the approaches to user.

It is necessary to quantify the effectiveness of algorithms for gaining knowledge about how correct unclear result we have got by using model of selecting features[5] and keyword diversification algorithms. So first measure is time needed for execution of the algorithms. Secondly queries appearing in result. In which we evaluate similarity between the search results we have as reported by top condition which we have given in each algorithm.

We utilize queries from the result to compare the result, by the accordance of the queries in result. In addition, we can determine the usefulness of result. We have gradually increased the complexity of input by considering distance calculation then singular words plural words and substring and lastly phonetic words still system responds efficiently.

The rest of the paper is organized as follows. Section II elaborates related work. Implementation details are presented along with mathematical modeling and experimental setup in section III and IV and V respectively. Experimental results are presented in section VI. Concluding remarks are presented in section VII.

II. RELATED WORK

In reference of Xrank [3], it accepts both XML & HTML documents and by using ranking scheme similar to PageRank scheme of Google they have evaluated ranked results. XRANK decides XML structure like hierarchy & hyperlink and A 2-D (two-dimensional) notion of keyword proximity, while computing the ranking for XML queries.

XRANK a generalizing system that derives the basic keywords from the Hyperlinks searched in search results of search engines XRANK can query over a mix of HTML & XML documents. But in paper they have currently taken a view i.e. document-centric, where they consider that query results are strictly hierarchical. We have to focus some parts like computing cost, unclear and repetitive result caused ranking scheme.
### Table 1. LITERATURE SURVEY

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Paper Title</th>
<th>Techniques and Methods Used</th>
<th>Findings</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Xrank: Rank keyword search over xml documents [1]</td>
<td>ElemRank, Hybrid Dewey Inverted List</td>
<td>Accept both XML and HTML documents, Only Document Centric Assumption of query results being strictly hierarchical</td>
</tr>
<tr>
<td>3</td>
<td>Processing xml keyword search by constructing effective structured queries [4]</td>
<td>Probabilistic model</td>
<td>Considered only structured database, Used method to balance relevance and novelty of keyword search, Probabilistic model of re-ranking</td>
</tr>
<tr>
<td>4</td>
<td>Diversifying search results [2]</td>
<td>Scoring function</td>
<td>Adaptive XML keyword search, Derive semantics of keyword query, Scoring function</td>
</tr>
</tbody>
</table>

A tabular format is analyzed to provide the data for relationships in the DivQ approach[3] A order of parent/child tables contributes to its taxonomy. The external tables can store the actual values. They have used greedy approach for approximation of ambiguous queries. Whenever, taxonomy of information is present in the document, the problem like unclear queries may arrive. They present an approach to give diversifying results those points diminish the threat of un-satisfaction of the user. However, it is hard to get this useful taxonomy and query logs. In addition, the differentiated outcomes in IR are often formed at document levels. [3]

Construction of structured queries candidates relates the issue of intent based keyword query diversification. [6]. Their brief idea while considering processing xml keywords search by using structured queries only[4] is to first map each keyword to a group of attributes (metadata), and then construct a giant number of structured query applicants by grouping the attribute-keyword pairs.

They assume that each structured query applicant represents a type of search motive i.e. a query interpretation. However, this works hard to be applied in real application caused by the following limitations: It may generate & This is done by analyzing keyword query & XML schemas. Limitation of this is metadata information of XML is responsible for constructing structured queries procedure & in XBridge giant number of queries get generated & evaluated so overhead of query logs and it Work only on structural constraints [7].

### III. IMPLEMENTATION DETAILS

![Diagram](image)

**A. Baseline Solution [10]—**

1. Load Dataset.
2. Enter Search Query.
3. Extract the sentences which contains search query.
4. Index the words to catch left and right word.
5. Apply Mutual score calculation.
6. Store score in key-value pair in ascending order.
7. Arrange scores in binary tree format.
8. Remove the repeated scores if leaf node.

By considering the example of search query ‘parallel’ we consider all the titles which contains word ‘parallel’ as input set. We calculate the Mutual information score for each word in titles by using following model [10]

\[
MI(x,y,T) = \text{Prob}(x,y,T) \times \log \left( \frac{\text{Prob}(x,y,T)}{\text{Prob}(x,T) \times \text{Prob}(y,T)} \right)
\]  

(1)

This considers probability of \( x \) in total input set \( T \) and probability of \( y \) in total input set \( T \). mutual score model defines if \( x \) and \( y \) are independent then knowing \( x \) does not give any information about \( y \) and vice versa that is its MI score is zero hence we can remove redundant terms. [10,5]

Then arranging all mutual scores in binary format we can decide root and leaf nodes and remove the repeated term in leaf if any. Output gives us the keywords which are nearer by MI score that is these keywords are contextually similar to each other.
B. Anchor pruning approach[10]–

1. Load Dataset.
2. Enter Search Query.
3. Extract the sentences, which contains search query.
4. Index the words to catch left and right word.
5. Apply Mutual score calculation.
6. Store score in key-value pair in ascending order.
7. Arrange scores in binary tree format.
8. Find smallest lowest common ancestor.
9. Remove the repeated scores if leaf node.
10. Go to step 2 for every associated query and repeat till step 9.

If search query used in baseline solution is ‘parallel’, and candidate queries we got are ‘systems’ and ‘computing’ then anchor pruning approach recursively iterate the steps by considering search query as ‘systems’ and search query as ‘computing’ this removes the least mutual score candidate queries and we can get more specific context.

C. Naïve based mapping

2. Calculate Levenshtein distance for search string
3. Select entries according to distance calculation from whole dataset.
4. For each selected entry
   - Calculate probability
5. Apply Naïve Bayes
6. Analyze on the basis of labeling done through .model file.

In past we are considering word ‘parallel’ as search query the input set is missing the words like[8] ‘parallelism’, ‘paralel’, ‘queries’ and substrings for parallel or “querree” for query, plural words like ‘queries’ and substrings like ‘parallelogram’ so we have gradually increasing the complexity of input dataset. Then we find occurrence of left and right context and then consider the depth wise candidates to analyze context similarity. For example if ‘database’ is contextually similar to ‘parallel’ then ‘parallel’ is similar to ‘processing’ so likelihood is to have database and processing comes under same group.

IV. MATHEMATICAL MODELLING

- U is main set of data about publications. U = {b1, b2, b3…….bn}
- S is a main candidate set queries.
  - S= {s1, s2, s3……sn}
- R is candidate set queries as reported by their mutual score or occurrence.
  - R= {r1, r2, r3……rn}
- D is set of Top qualified queries.
  - D= {d1, d2, d3……dn} this set is of top qualified queries.
- Set n to be the length of s, Set m to be the length of t.
  - If n = 0, return m and exit. If m = 0, return n and exit.
  - Construct a matrix containing 0.m rows and 0..n columns.
  - Initialize the first row to 0..n. Initialize the first column to 0..m.
  - Examine each character of s (i from 1 to n), examine each character of t (j from 1 to m).
  - If s[i] equals t[j], the cost is 0. If s[i] doesn’t equal t[j], the cost is 1.
- Set cell d[i,j] of the matrix equal to the minimum of:
  - b. The cell immediately to the left plus 1: d[i,j-1] + 1.
  - c. The cell diagonally above and to the left plus the cost: d[i-1,j-1] + cost.
  - After the iteration steps the distance is found in cell d[n,m].

- Mutual Information Model[10,5]

\[ \text{MI}(x,y,T) = \text{Prob}(x,y,T) \times \log \left( \frac{\text{Prob}(x,T) \times \text{Prob}(y,T)}{\text{Prob}(x,y,T)} \right) \]

- Probabilistic model

Result = Total Current Query Occurrence Count / Total Result Size
Prob_count = log(result)

V. EXPERIMENTAL SETUP

The system is developed using Java framework (version jdk 8) on Windows platform. The eclipse (version 8.1) is used as a development tool. Weka tool has been used for the generating .model file for mapping of labeling of similar words. The system doesn’t need any specific hardware to run any standard machine (with processor next to P4) is able of running the application.
VI. RESULTS AND DISCUSSION

System considers DBLP database. Figure 2 demonstrates the time comparison graph between baseline, anchor pruning, mapped based naïve and advanced filtering algorithms. In proposed system that is third and fourth approach time required is better as compared with the previous system. Figure 3 demonstrates the sequence of results which defines usefulness of search queries. That is basically dependent on intension of user which he or she wants to search. We gradually increase the dataset complexity by considering distance between letters, plural, substring so we are improving efficiency.

VII. CONCLUSION

After considering four different approaches like baseline, anchor pruning, naïve based mapping and advanced filtering outcome become more accurate. We are defining context similarity through different methods. Baseline solution suggests mutual information model brings you similarity-based results. Anchor pruning claims to be more nearer to domain of search query by recursively process the associated result queries. Naïve approach is based on the concept of labeling the data in advance to get result early. And advanced filtering handles singular, plural, substring and some phonetic words that is we moderately adds complexity to input data and still tries to get outcome fast and accurate. Efficiency is calculated as per input size and Accuracy is calculated as per complexity of input data. Here we have used DBLP database which includes book related data including many words and group of some words specify particular domain i.e. context, going level wise in depth of occurrence, similarity we conclude to get usefulness of candidate queries which can be justified as per users intension.

V. REFERENCES

[10] Jianxin Li, Chengfei Liu, Member, IEEE, and Jeffrey Xu Yu, Senior Member, IEEE, “Context-Based Diversification for Keyword Queries Over XML Data,”IEEE transactions on knowledge and data engineering.VOL. 27, NO. 3, MARCH 2015