A Comparative Study of MySQL Functions for XML Element Retrieval

Chuleerat Jaruskulchai, Member, IAENG, and Tanakorn Wichaiwong, Member, IAENG

Abstract—Due to the ever increasing information available electronically, their size is growing rapidly. Since, XML documents have further information; document representation of these might be changed. Term weight technique is a mechanism to retrieve documents. In XML element retrieval applications, the weighting algorithm plays an important role and it greatly affects the precision and recall results of the retrieval systems. MySQL’s full text search algorithm is widely applied into retrieval flat document. However, for the semi-structure document hasn’t evaluation. In this paper, we have to investigate the weighting functions of MySQL and performed a comparative study of weighting schemes processing. Our objective of the study was to find out the appropriate features to achieve the effectiveness of XML element retrieval. The experiment results show the Natural Language use of vector space model function performs better than other methods measured by INEX evaluations.

Index Terms—MySQL Full Text Search, XML Retrieval, Ranking Strategies.

I. INTRODUCTION

The weighting functions of information retrieval [1], [2] have been widely studied. Many researchers have been studied base on different weighting functions. We developed XML [3] information retrieval system by using MySQL [4]. For this purpose, our study addressing on the comparison of various term weighting base on MySQL’s features and contribution of weighting schemes area of understanding features that influence the automatic indexing potential of terms.

This paper is organized as follows; Section 2 reviews related works. Section 3 explains our experiments, conclusions and further work are drawn in Section 4.

II. RELATED WORKS

A. MySQL Full Text Search Overview

MySQL Full Text Search (FTS) [4] has multiple table types; MyISAM has support for an FTS index. It is implemented differently than other FTS systems. Instead of storing the inverted lists in a tightly packed binary format, it uses a normal two-level B-Tree. The first level contains records of the form (word, count), where the count is the number of documents in which the word appears. The second level contains records of the form (weight, rowID), where weight is a floating point value signifying the relative importance of the word in the document pointed to by rowID. The full text search support in MySQL uses the following constructs.

- **MATCH** takes a comma-separated list that names the columns to be searched.
- **AGAINST** takes a string to search for and an optional modifier that indicates what type of search to perform.

The search string must be a literal string rather than a variable or a column name

MySQL has three types of weighting functions:

- **Natural Language Searches**: By default, the MATCH function performs a natural language search for a string against a text collection. A collection is a set of columns included in FULLTEXT indices. The search string is given as the argument to AGAINST function. For each row in the table, MATCH function returns a relevance value; that is, a similarity measure between the search string and the context. When MATCH function is used in a WHERE clause, as in the example shown earlier, the rows returned are automatically sorted with the highest relevance first. Relevance is computed based on the number of words in the row, the number of unique words in that row, the total number of words in the collection, and the number of documents that contain a word. The first query sorts the results by relevance whereas the second does not. However, the second query performs a full table scan and the first does not. The first may be faster if the search matches few rows; otherwise, the second may be faster because it would read many rows anyway. For example, although the word “mysql database” is present in every row in the column “details” of “inex08” table shown earlier as below;

```sql
SELECT xPath, MATCH (details) AGAINST ('mysql database' IN NATURAL LANGUAGE MODE) AS score FROM inex08 WHERE MATCH (details) AGAINST ('mysql database' IN NATURAL LANGUAGE MODE);
```

- **Boolean Searches**: MySQL can perform boolean searches using the IN BOOLEAN MODE modifier. With this modifier, certain characters have special meaning at the beginning or end of words in the search string. In the following query, the + and - operators indicate that a word is required to be present or absent,
respectively, for a match to occur. In implementing this feature, MySQL uses what is sometimes referred to as
implied boolean logic, in which
1) + stands for AND
2) − stands for NOT
3) [no-operator] stands for OR
For example, the query retrieves all the rows that contain the word “mysql” but that do not contain the word “database” is present in every row in the column “details” of “inex08” table shown earlier as below;

\[
\text{SELECT xPath FROM inex08 WHERE MATCH (details) AGAINST ('+mysql -database' IN BOOLEAN MODE);
}\]

- **Query Expansion**: FTS supports query expansion (and in particular, its variant “blind query expansion”). This is generally useful when a search phrase is too short, which often means that the user is relying on implied knowledge that the full-text search engine lacks. For example, a user searching for “database” may really mean that “MySQL”, “Oracle”, “DB2”, and “RDBMS” all are phrases that should match “databases” and should be returned, too. This is implied knowledge. Blind query expansion (also known as automatic relevance feedback) is enabled by adding WITH QUERY EXPANSION following the search phrase. It works by performing the search twice, where the search phrase for the second search is the original search phrase concatenated with the few most highly relevant documents from the first search. Thus, if one of these documents contains the word “databases” and the word “MySQL”, the second search finds the documents that contain the word “MySQL” even if they do not contain the word “database”. Because blind query expansion tends to increase noise significantly by returning no relevant documents, it is meaningful to use only when a search phrase is rather short.

\[
\text{SELECT xPath FROM inex08 WHERE MATCH (details) AGAINST ('database' WITH QUERY EXPANSION);
}\]

The element scoring of its context using vector space model of MySQL-FTS as following:

\[
\text{LeafScore}(e, Q) = \sum_{t \in Q} Q_t \times W_t \times L_t
\]

\[
L_t = \log\left(\frac{tf_t + 1}{len(e)}\right) \times \frac{U}{U + 0.0115 \times U}
\]

\[
W_t = \log\left(\frac{N - e_t}{e_t}\right)
\]

\[
Q_t = Q_t f_t
\]

Note that;
LeafScore(e, Q) measures the relevance of element e to a query Q.
\(W_t\) is the inverse element frequency weight of a term t.
\(tf_t\) is the frequency of a term t occurring in an element e.
\(len(e)\) is the length of an element e.

\[
U = \text{the number of unique terms in element } e.
N = \text{the total number of an element in the collection.}
e_t = \text{the total element of a term } t\text{ occur.}
Q_t f_t = \text{the frequency of a term } t\text{ occurring in a query } Q.
\]

**B. The Score Propagation**

Score Propagation [5], [6], [7] is used to rank elements based on leaf-node indexing. Scoring is propagated upward to ancestors. The resulting relevance score for each element is a weighted accumulation of ranking scores of an element’s children. This strategy was presented by the Gardens Point XML-IR (GPX) [5], [6], [7], a propagation method that was proposed as a bottom-up scheme (BUS) [8]. For example, for each element with only one relevant child element, the child should be ranked higher. Otherwise, this element ranks higher than their child. The assignment of a numeric score to a document given a query can be represented as follows:

\[
score(e, q) = D(m) \times \sum_{e, c} \text{score}(e, c, q)
\]

Note that;
\(D(m)\) is the smoothing parameter set as follows.
If e has one child, then \(D(m) = 0.49\).
Otherwise, \(D(m) = 0.99\).

**C. XML System Overview**

Our system uses a relational database as a storage back end and query processing methods are based on Full-Text Search. In the following, we discuss the schema setup using MySQL [4] engine generally available release: 5.1.51.

In Fig. 1, depicts the overview of XML retrieval system. For the initial step, we consider a simplified XML data model, but disregarding any kind of markup including comments, link and attributes. The main components of the XML retrieval system are including:

- The Absolute Document XPath Indexing (ADXPI) [9], when new documents are entered, the indexer parses and analyzes the tag and content data to build the list of leaf nodes.
- The score sharing method, which allows assigning the parent scores by sharing score from leaf node to their parents by a top down scheme approach.
D. The Score Sharing Function

In a previous study [10], we compute the scores of all elements in the collection that contain query terms. We consider the scores of elements by accounting for their relevant descendants. The scores of retrieved elements are now shared between the leaf node and their parents in the relevant descendants. The scores of retrieved elements are considered by accounting for their elements in the collection that contain query terms. We compute the scores of all elements in the collection that contain query terms. We consider the scores of elements by accounting for their relevant descendants. The scores of retrieved elements are now shared between the leaf node and their parents in the relevant descendants. The scores of retrieved elements are considered by accounting for their elements in the collection that contain query terms.

\[
\text{Score}(\text{PNode}) \leftarrow \text{Score}(\text{PNode}) + \langle \text{LeafScore} \times \beta^n \rangle
\]

Note that;

\[\text{PNode}\] is a current parent node.

\[\beta\] is a tuning parameter.

\[IF(0 - 1) THEN\] preference is given to the leaf node over the parents.

\[OTHERWISE\] preference is given to the parents.

\[n\] is the distance between the current parent node and the leaf node.

E. The ADXPI Indexer

According to previous studies [9], a single inverted file can hold the entire reference list, while a suitable indexing of terms can support the fast retrieval of term-inverted lists. To control for overlap and reduce the cost of joined on relational database; we use the ADXPI scheme to transform each leaf element level into a document level. For instance, take a document named "x1" as shown in Fig.2.

Fig. 2, depicts the example of the XML element trees then we can build an index by ADXPI expression identifies a leaf node that has text contain within the document, relative to document and their parents are following:

- x1/article[1]/title[1]: "xml"
- x1/article[1]/body[1]: "xml"
- x1/article[1]/body[1]/section[1]: "retrieval"
- x1/article[1]/body[1]/section[1]/title[1]: "xml"
- x1/article[1]/body[1]/section[1]/p[1]: "information"
- x1/article[1]/body[1]/section[1]/p[2]: "retrieval"

III. EXPERIMENT SETUP

In this section, we present and discuss the results based on the INEX collection. We also present the results of an empirical sensitivity analysis of various parameter performed on a Wikipedia collection. This experiment was performed on Intel Pentium i5 4 * 2.79 GHz with 6 GB of memory, Microsoft Windows 7 Ultimate 64-bit Operating System and Microsoft Visual C♯.NET 2008.

A. Data sets


B. INEX Evaluations

As for INEX-IEEE effectiveness [11], we refer to the relative and absolute precision values as well as the non-interpolated mean average precision (MAP), which displays absolute precision as a function of absolute recall using official relevance assessments provided by INEX. Furthermore, the following, more sophisticated and XML-specific metrics were newly introduced for the INEX-IEEE benchmark: nxCG. The normalized extended Cumulated Gain (CG) metrics are an extension of the Cumulated Gain metrics that consider the dependency of XML elements (e.g., overlap and near-misses) within an evaluation.

As for INEX-Wikipedia effectiveness [13], we refer to the main ranking of INEX competition based on iP[0.01] instead of the overall measure MAiP, allowing us to emphasize precision at low recall levels. Our experiment targets Content Only (CO) Task only as well as systems that accept CO queries. Note that CO queries are terms enclosed in the <title> tag. Then, only the Focused Task remains in the INEX during the period 2005-2008. Thus, the system is evaluated only using Focused Task according to the inexact eval and Eval tools provided by INEX.

C. Experiment Results and Discussion

In this section, we tuned parameters using INEX-2005 Adhoc track evaluation scripts distributed by the INEX organizers. Our tuning approach was such that the sums of all relevance scores are maximized. The total number of leaf node is 2500 and the \(\beta\) parameter is set to 0.10 [10], which is used to compute the sharing score function.

In order to evaluate the sensitivity of the evaluation, we have used the entire MySQL features are including Natural Language Searches (NLS), Boolean Searches (BLS), Query Expansion (QE), and Natural Language Searches with Query Expansion (NLSQE). As such, we report the effectiveness of our system on INEX collections as follows:

The performance of different feature and ranking models is evaluated. Tables I, II, III and Fig. 3, 4, 5 on INEX-Wikipedia collection and Tables IV, V and Fig. 6, 7 show the comparison of effectiveness to GPX system more details are following:

The run NLS obtained the highest scores for INEX-Wikipedia on 2006 topics is 0.4518, 0.2124 for BLS, 0.3163 for QE, and 0.3389 for NLSQE at iP[0.01] respectively. The
TABLE I
THE EFFECTIVENESS ON INEX-2006 FOCUSED TASK

<table>
<thead>
<tr>
<th>RunID</th>
<th>iP[0.00]</th>
<th>iP[0.01]</th>
<th>iP[0.05]</th>
<th>iP[0.10]</th>
<th>MAiP</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSL</td>
<td>0.4539</td>
<td>0.4518</td>
<td>0.4044</td>
<td>0.3651</td>
<td>0.1109</td>
</tr>
<tr>
<td>BLS</td>
<td>0.2657</td>
<td>0.2124</td>
<td>0.1193</td>
<td>0.0773</td>
<td>0.0257</td>
</tr>
<tr>
<td>QE</td>
<td>0.3177</td>
<td>0.3163</td>
<td>0.2831</td>
<td>0.2556</td>
<td>0.0776</td>
</tr>
<tr>
<td>NLSQE</td>
<td>0.3404</td>
<td>0.3389</td>
<td>0.3033</td>
<td>0.2738</td>
<td>0.0832</td>
</tr>
</tbody>
</table>

TABLE II
THE EFFECTIVENESS ON INEX-2007 FOCUSED TASK

<table>
<thead>
<tr>
<th>RunID</th>
<th>iP[0.00]</th>
<th>iP[0.01]</th>
<th>iP[0.05]</th>
<th>iP[0.10]</th>
<th>MAiP</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSL</td>
<td>0.4800</td>
<td>0.4169</td>
<td>0.3186</td>
<td>0.2539</td>
<td>0.0987</td>
</tr>
<tr>
<td>BLS</td>
<td>0.1605</td>
<td>0.1349</td>
<td>0.0733</td>
<td>0.0500</td>
<td>0.0179</td>
</tr>
<tr>
<td>QE</td>
<td>0.3936</td>
<td>0.3419</td>
<td>0.2613</td>
<td>0.2082</td>
<td>0.0809</td>
</tr>
<tr>
<td>NLSQE</td>
<td>0.3408</td>
<td>0.2960</td>
<td>0.2262</td>
<td>0.1803</td>
<td>0.0701</td>
</tr>
</tbody>
</table>

TABLE III
THE EFFECTIVENESS ON INEX-2008 FOCUSED TASK

<table>
<thead>
<tr>
<th>RunID</th>
<th>iP[0.00]</th>
<th>iP[0.01]</th>
<th>iP[0.05]</th>
<th>iP[0.10]</th>
<th>MAiP</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSL</td>
<td>0.6838</td>
<td>0.5740</td>
<td>0.4262</td>
<td>0.3411</td>
<td>0.1187</td>
</tr>
<tr>
<td>BLS</td>
<td>0.3338</td>
<td>0.3008</td>
<td>0.1859</td>
<td>0.1371</td>
<td>0.0584</td>
</tr>
<tr>
<td>QE</td>
<td>0.5812</td>
<td>0.4879</td>
<td>0.3623</td>
<td>0.2899</td>
<td>0.1009</td>
</tr>
<tr>
<td>NLSQE</td>
<td>0.5129</td>
<td>0.4305</td>
<td>0.3197</td>
<td>0.2558</td>
<td>0.0890</td>
</tr>
</tbody>
</table>

TABLE IV
COMPARE TO GPX IN THE INEX-2007 AD HOC TRACK

<table>
<thead>
<tr>
<th>RunID</th>
<th>iP[0.00]</th>
<th>iP[0.01]</th>
<th>iP[0.05]</th>
<th>iP[0.10]</th>
<th>MAiP</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLS</td>
<td>0.4800</td>
<td>0.4169</td>
<td>0.3186</td>
<td>0.2539</td>
<td>0.0987</td>
</tr>
<tr>
<td>GPX</td>
<td>0.4086</td>
<td>0.3842</td>
<td>0.3433</td>
<td>0.3208</td>
<td>0.1541</td>
</tr>
</tbody>
</table>

TABLE V
COMPARE TO GPX IN THE INEX-2008 AD HOC TRACK

<table>
<thead>
<tr>
<th>RunID</th>
<th>iP[0.00]</th>
<th>iP[0.01]</th>
<th>iP[0.05]</th>
<th>iP[0.10]</th>
<th>MAiP</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLS</td>
<td>0.6838</td>
<td>0.5740</td>
<td>0.4262</td>
<td>0.3411</td>
<td>0.1187</td>
</tr>
<tr>
<td>GPX</td>
<td>0.6818</td>
<td>0.6344</td>
<td>0.5693</td>
<td>0.5178</td>
<td>0.2587</td>
</tr>
</tbody>
</table>

run NLS obtained the highest scores for INEX-Wikipedia on 2007 topics is 0.4169, 0.1349 for BLS, 0.3419 for QE, and 0.2960 for NLSQE at iP[0.01] respectively. The run NLS obtained the highest scores for INEX-Wikipedia on 2008 topics is 0.5740, 0.3008 for BLS, 0.4879 for QE, and 0.4305 for NLSQE at iP[0.01] respectively.

IV. CONCLUSION

Due to the ever increasing information available electronically, their size is growing rapidly. The widespread use of XML documents in digital libraries led to the development of information retrieval (IR) methods specifically designed for XML collections. Most traditional IR systems are limited to whole document retrieval; however, since XML documents separate content and structure, XML-IR systems are able to retrieve the relevant portions of documents. Therefore, users who utilize an XML-IR system could potentially receive highly relevant and highly relevant and precise material.

In this paper, we have to investigate the weighting functions of MySQL and performed a comparative study of weighting schemes processing. Our objective of the study was to find out the appropriate features to achieve the effectiveness of XML element retrieval. Our experiment shows
that the Natural Language search function used is the TF-IDF performs better than other methods measured by INEX evaluations on iP[0.01] and MAiP.

In our future work, we plan to study how to make inferences regarding structural aspects based on Content and Structure (CAS) queries.

REFERENCES