International Journal For Technological Research In Engineering
Volume 1, Issue 10, June-2014 *ISSN (Online): 2347 - 4718*

TERRIER TOOL EXPERIMENTS: ROMAN, DEVANAGARI, MIXED SCRIPTS IN INFORMATION RETRIEVAL

Honey Patel[1] (M.E Student), Krunal Panchal[2] (Asst Prof)
Computer Engineering Department,
L.J Institute of Engineering and Technology
Ahmedabad, Gujarat, India

Abstract: In this paper, we describe Terrier, solve query drift problem and existing new approach apply on search engine that allows the rapid development of large-scale retrieval applications. We focus on the open source version of the software, which provides a comprehensive, flexible, robust, and transparent test-bed platform for research and experimentation in Information Retrieval.
Keywords: Terrier, Information Retrieval, TF-IDF Model

I. INTRODUCTION

Information retrieval is the activity of obtaining information resources relevant to an information need from a collection of information resources. An information retrieval process begins when a user enters a query into the system. Queries are formal statements of information needs, for example search strings in web search engines. In information retrieval a query does not uniquely identify a single object in the collection. Instead, several objects may match the query, perhaps with different degrees of relevancy. It does not explicitly return information or answer questions. Instead, it informs on the existence and location of documents that might contain the desired information. Some suggested documents will, hopefully, satisfy the user's information need. These documents are called relevant documents. Many university, corporate, and public libraries now use IR systems to provide access to books, journals, and other documents. Commercial IR systems offer databases containing millions of documents in myriad subject areas. Dictionary and encyclopedia databases are now widely available for PCs. IR has been found useful in such disparate areas such as office automation and software engineering. Indeed, any discipline that relies on documents to do its work could potentially use and benefit from IR.

A. Indexing

Show Figure outlines the indexing steps in Terrier. Indexing is a four step procedure and at each Stage plug-ins can be added for customization. These five steps are:

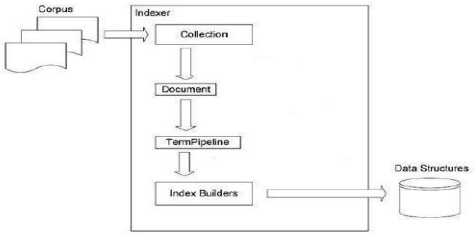

Figure 1: Indexing Architecture of Terrier

- Extraction of Document objects from Collections generated from Corpus input.
- Collection reads documents.
- Each Document for extracting terms (basic indexed unit) and related info.
- Term Processing using Term Pipelines like stemmer.
- Index Building.

The indexing can be configured by changing appropriate properties in etc/terrier.properties. Each stage mentioned above is associated with some properties. Terrier comes with various document parsers embedded. These include the ability to index HTML documents, Plain text documents, Microsoft Word, Excel and PowerPoint documents, as well as Adobe PDF files. To add support for another file format to Terrier, a developer must only add another Document plug-in which is able to extract the terms from the document [4].

B. Retrieval

Figure 1.3 provides an outline of the retrieval process in Terrier. Terrier offers great flexibility in choosing a weighting model, as well as in altering the score of the retrieved documents. Another very important retrieval feature of Terrier is the automatic query expansion.

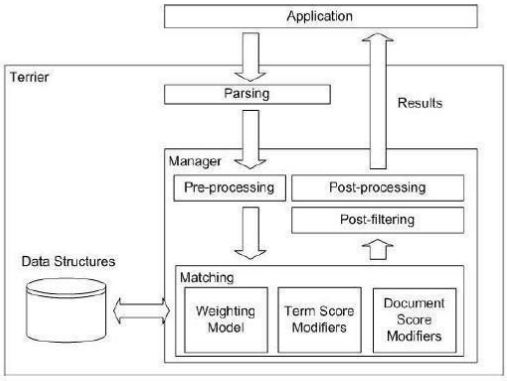

Figure 2: Architecture of Retrieval

In Retrieval side, Theses nine steps are:

- It is the input which the application provides to Terrier. We are using Multi-term Query.
- Parse the query in Manager.
- Each query term is processed via Term Pipeline.
- In matching Section, The Weighting model represents the retrieval model that is used to weight the terms of a document. The weighting models are

like to TF-IDF, BM25, DFR (Divergence from Randomness).

- Document score modifiers are responsible for query dependent modification of document scores.
- Term score modifiers modify the scores of the documents for a given set of postings.
- Matching modifiers compares with data structure (corpus).
- Post filter modules filter the result set.
- Post process modules alter the result set and give to application.

II. TF-IDF MODEL

- Tf-idf stands for term frequency-inverse document frequency.
- tf-idf weight is a weight often used in information retrieval .
- tf-idf weighting scheme are often used by tool in scoring and ranking a document's relevance given a user query.
- A document with 10 occurrences of the term may be more relevant than a document with one occurrence of the term.
- But not 10 times more relevant.
- df t= document frequency for term t idft= inverse document frequency of term t, idft = log2 (N/ df t) (N: total number of documents)
- tft,d = term frequency (how often term t occurs in document d)
- tf-idft,d = a combined weight for term t in document d tf- idft,d = tft,d * idft

$$score(q,d) = \sum_{t \in q} tf\text{-}idf_{t,d} \quad (1)$$

III. QUERY NORMALIZATION TO TERMS

We need to "Normalize" words in indexed text as well s query words into the same form.

We want to match like "sapano.n" and "sapanon both are same meaning".

Results is terms:a term is a normalized word type which is an entry in our IR system dictionary.

We most commonly implicitly define equivalence classes of terms by,e.g.,

Deleting periods to form a term like "mere , mere" and "chaa.nd" etc.

Deleting hyphens to form a term like "jaa_e","pyaar,hua" etc.

Same meaning name but spellings are different e.g. ,"sapnoo","sapnoon"

IV. QUERY EXPANSION

Terrier includes automatic pseudo relevance feedback, in the form of Query Expansion (QE). The method works by taking the top most informative terms from the top-ranked documents of the query, and adding these new related terms into the query. This operation is made possible by the presence of the direct index. The direct index allows the terms and their frequencies to be determined for each document in the index. The new query is reweighted and rerun providing a richer set of retrieved documents. In addition, Terrier would increase a recall values with the help of Rocchio's method . Automatic query expansion is highly effective for many IR tasks. Query expansion E.g., Query is "mujhko saza di pyaar ki" so, Expanded Query is "mujhko saza di pyaar ki mujh ko piyaar.

V. SUBSTITUTE SYNONYM

This operation replaces synonym in reference words.In substitute synonym ,a word have been more than one words meaning in documents.

Example: "raat" and "raatari" both are same meaning.

Raat -> raatari
Tum -> tujh,
tere Jiya -> jiyu
Tumse ->tum
Zindagi->jiivan ,zindagaani

VI. CASE FOLDING

Case folding is a reduce all letters to upper case.
So,If any type of words like mid letters and last letters have in upper case than convert into lower case Example:

chaaNd => chaand jahaaN => jahaan dapraN => dapran deewaaNa => deewaana

VII. PORTER"S ALGORITHM

Porter"s algorithm is a commonest algorithm for stemmer. Results suggest it"s at least as good as other stemming option. In porter"s algorithm, reduce phases sequentially, each phase consist set of command.

E.g., raanii=>raani
Typically rules in porter:-
=> i aa =>a oo =>u hh=>h
„a" occurrence more than 2 times =>aa
„e" occurrence more than 2 times=>ee

VIII. QUERIES AND DOCUMENTS

I have 25 queries provided by FIRE community.

The queries are bollywood song lyrics. The documents have 62,888 which contained songs and lyrics in Roman (ITRANS or plain format), Devanagari and mixed scripts.

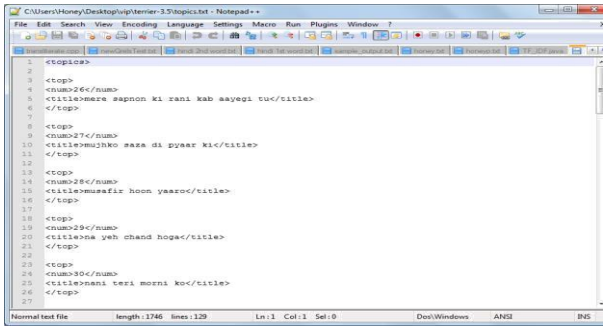

Figure 3: Queries

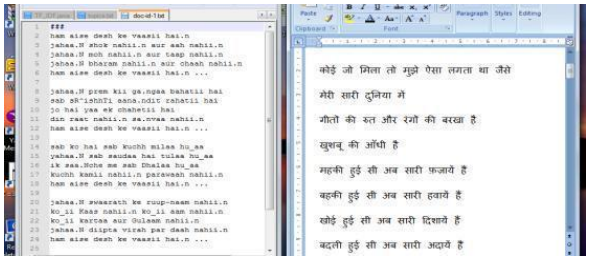

Figure 4: Bollywood lyrics songs

IX. EVALUATION MATRICS

$$AveP = \frac{\sum_{k=1}^{n}(P(k) \times rel(k))}{\text{No. of relevant documents}} \quad (2)$$

Where k is the rank in the sequence of retrieved documents, n is the number of retrieved documents, P(k) is the precision at cut-off k in the list and relk is an indicator function equaling 1 if the item at rank k is a relevant document, zero otherwise. Then,

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} \quad (3)$$

Where Q is the number of queries. In our case, we consider relevance judgments 1 and 2 as non-relevant, and 3, 4, and 5 as relevant. MAP was computed after looking at the first ten retrieved documents. The reciprocal rank of a query response is the multiplicative inverse of the rank of the first correct answer (ranki). MRR is the average of the reciprocal ranks of results for a sample of queries Q

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i} \quad (4)$$

In our case, we consider relevance judgments 1 and 2 as incorrect answers, and 3, 4 and 5 as correct answers.MRR was computed after looking at the first ten retrieved documents. We observed that minor changes in these conventions do not alter the general trends of the results.

X. RESULTS AND COMPARISON

	OLD VALUES		NEW VALUES
MAP	0.255	1.896	
MRR	0.584	0.39	

Table 1: Old values using TF-IDF model and New Values using New Existing Approach

XI. MAP V/S MRR Graph

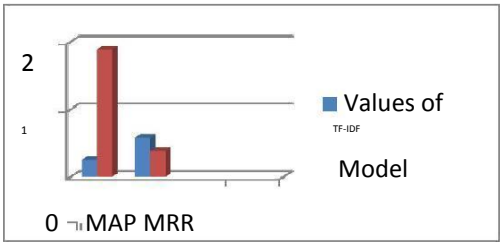

Figure 5: MAP V/s MRR results of 25 query for top ten relevant documents.

XII. CONCLUSION

I concluded in this paper improve the precision values but not improve the Recall values. So, it has increased the MAP values and Decrease the MRR values. Precision and Recall is vice-versa. So, in same time precision and recall values are not increasing or decreasing.

REFERENCES

[1] Parantapa Goswami ", Batch (TREC) Indexing and Retrieval using Terrier 2.2.1", Indian Statistical Institute, Kolkata.
[2] Craig Macdonald, Iadh Ounis," Expertise Drift and Query Expansion in Expert Search", University of Glasgow Glasgow, Scotland, UK.
[3] www.isical.ac.in
[4] http://terrier.org

