Novel Document Retrieval Algorithm for Unstructured Data

[1]Pragya Udaypure, [2]Ajay Phulre,
Department of Computer Science & Engineering ,
Shri Balaji Institute of Technology and Management ,
Betul, MP, India
Email – [1]udaypureybitti@gmail.com, [2] aphulre@gmail.com

Abstract: *Nowadays a common size of document rendering over internet might have more than 350 billion pages. It is almost impossible for a reader to read thought all documents and find out relative information in a couple of minutes. The growth of the Internet and the availability of enormous volumes of data in digital form have necessitated intense interest in techniques to assist the user in locating data of interest. The Internet has over million and billion pages of data and is expected to increase day by day. A large quantity of information the average person does not care about is buried over the internet. The Digital Library effort is also progressing, with the goal of migrating from the traditional book environment to a digital library environment.Thus the importance of text retrieval systems has grown dramatically during recent years due to very rapid increase of available storage capacity, increased performance of all types of processors and exponential growth of the global networks that provide an enormous source of different documents.*

Key Words: *Document retrieval, Text Retrieval, Indexing*

1.INTRODUCTION

Nowadays, a large quantity of data is being accumulated. Usually there is a huge gap from the stored data to the knowledge that could be interpreted from the data. This transition won't occur automatically, that's why mining of data is required. In Exploratory Data Analysis, some initial knowledge is known about the data, but Data Mining could help in a more in-depth knowledge about the data. Seeking knowledge from massive data is one of the most desired attributes of Data Mining. Manual data analysis has been around for some time now, but it creates a bottleneck for large data analysis. Fast developing computer science and engineering techniques and methodology generates new demands to mine complex types of data. A number of Data Mining techniques (such as association, clustering, classification) are developed to mine this vast amount of data. Previous studies of Data Mining mostly focused on structured data, such as relational, transactional and data warehouse data. However, in reality, a substantial portion of the available information is stored in text databases (or document databases), which consists of large collections of documents from various sources, such as news articles, books, digital libraries and Web pages. Text databases are rapidly growing due to the increasing amount of information available in electronic forms, such as electronic publications, e-mail, CD-ROMs, and the World Wide Web (which can also be viewed as a huge, interconnected, dynamic text database).

Data stored in most text databases are semi structured data in that they are neither completely unstructured nor completely structured. For example, a document may contain a few structured fields, such as title, authors, publication date, length, category, and, so on, but also contain some largely unstructured text components, such as abstract and contents. Information Retrieval techniques, such as text indexing, have been developed to handle unstructured documents. But, traditional Information Retrieval techniques become inadequate for the increasingly vast amounts of text data. Typically, only a small fraction of the many available documents will be relevant to a given individual or user. Without knowing what could be in the documents, it is difficult to formulate effective queries for analyzing and extracting useful information from the data. Users need tools to compare different documents, rank the importance and relevance of the documents, or find patterns and trends across multiple documents. Thus, Text Mining has become an increasingly popular and essential theme in Data Mining.

1.1 Data Mining – Concepts and Techniques

Database technology has evolved from primitive file processing to the development of database management systems with query and transaction processing. Due to the explosive growth in data collected from applications including business and management, government administration, scientific and engineering, and environmental control, there is an increasing demand for efficient and effective data analysis and data understanding tools. Data warehouse systems provide some data analysis capabilities which include data cleaning, data integration and OLAP (On-Line Analytical Processing). These analysis techniques provide functionalities such as summarization, consolidation and aggregation, as well as the ability to view information at different angles. Although OLAP tools support multidimensional analysis and decision making, additional data analysis tools are required for in-depth analysis, such as data classification, clustering, and the characterization of data changes over time. The widening gap

between data and information calls for a systematic development of Data Mining tools which will turn data tombs into "golden nuggets" of knowledge. Data mining tools perform data analysis which may uncover important data patterns, contributing greatly to business strategies, knowledge bases, and scientific and medical research.

2. LITERATURE REVIEW:

Since the number of references for this research project work is large, some references are summarized in this section. In the subsequent chapters, relevant literature is included directly in the main text.

K.Sreerama Murthy, et.al. [4] discussed some problems in conventional IR methods which are mentioned with proposed solutions. Conventional IR methods become insufficient to handle large Text Databases containing high quantity of text documents. To explore relevant documents from the large document collection, inverted files are used which are then read from the disk. The major cost of searching process are the space requirement in memory to hold inverted file entries, and the time spend to process huge size inverted files maintaining record of each document of the quantity as they are potential solutions. To speed up the procedure of text document retrieval and effective use of the memory space, They proposed an algorithm which is based on inverted index file. By using the range partition feature of oracle, the space requirement of memory is condensed considerably as the inverted index file is stored on secondary storage and only the required portion of the inverted index file is maintained in the main memory.

B. Ganga [6], in the process of document retrieval, focused on work to combine the advantages of two methodologies: suffix tree and Boyer-Moore. As a result the Phrase Based Document retrieval Algorithm represent each document as suffix trees, where phrases of the documents are stored as suffixes of tree in hash table for efficient storing and retrieval and Boyer Moore algorithm is used to check the presence of pattern i.e. the input phrase in order and without order. General description of the algorithm is, it takes input as set of documents in the form of Portable Document Format, MS Format files and Text files and input phrase and implements suffix tree algorithm which represent document as suffix and then Boyer-Moore algorithm is applied and output is obtained as the set of documents which contains the input phrase in order as well as without order.

Ning Zhong, et al. [11] put forward the somewhat negative part of phrase-based approach of searching. Although phrases are less ambiguous and more discriminative than individual terms, the likely reasons for the discouraging performance include:

1) Phrases have inferior statistical properties to terms,
2) They have low frequency of occurrence, and
3) There are large numbers of redundant and noisy phrases among them.

R. Sagayam, et al. [10] mentions two indexing techniques which are inverted indices and signature files. An inverted index is an index structure that maintains two hash indexed or B+-tree indexed tables: document table and term table, where document table consists of a set of document records, whereas term table consists of a set of term records. With such organization, it is easy to answer queries like "Find all of the documents associated with a given set of terms," or "Find all of the terms associated with a given set of documents". For example, to find all of the documents associated with a set of terms, we can first find a list of document identifiers in term table for each term, and then intersect them to obtain the set of relevant documents. Inverted indices are widely used in industry. They are easy to implement.

A signature file is a file that stores a signature record for each document in the database. Each signature has a fixed size of b bits representing terms. A simple encoding scheme goes as illustrated. Each bit of a document signature is initialized to 0.A bit is set to 1 if the term it represents appears in the document. A signature S1 matches another signature S2 if each bit that is set in signature S2 is also set in S1. Since there are usually more terms than available bits, multiple terms may be mapped into the same bit. Such multiple-to-one mappings make the search expensive because a document that matches the signature of a query does not necessarily contain the set of keywords of the query. The document has to be retrieved, parsed, stemmed, and checked. Improvements can be made by first performing frequency analysis, stemming, and by filtering stopwords, and then using a hashing technique and superimposed coding technique to encode the list of terms into bit representation. Nevertheless, the problem of multiple-to-one mappings still exists, which is the major disadvantage of this approach.

3. IMPLEMENTATION DETAILS:

This section covers the methodologies used and the implementation part of the research work. The brief introduction of the methodologies and the rules used in the project are given here

System Architecture

Figure 3.1 describes the overall architecture of the system where it shows the different component or the modules will be used in the proposed system.

Subtasks — components of a larger text-analytics effort — typically include:

- Information retrieval or identification of a corpus is a preparatory step: collecting or identifying a set of textual materials, on the Web or held in a file system, database, or content management system, for analysis.
- Although some text analytics systems apply exclusively advanced statistical methods, many others apply more extensive natural language processing, such as part of speech tagging, syntactic parsing, and other types of linguistic analysis.
- Named entity recognition is the use of gazetteers or statistical techniques to identify named text features: people, organizations, place names, stock ticker symbols, certain abbreviations, and so on. Disambiguation — the use of contextual clues — may be required to decide where, for instance, "Ford" can refer to a former U.S. president, a vehicle manufacturer, a movie star, a river crossing, or some other entity.
- Recognition of Pattern Identified Entities: Features such as telephone numbers, e-mail addresses, and quantities (with units) can be discerned via regular expression or other pattern matches.
- Coreference: identification of noun phrases and other terms that refer to the same object.
- Relationship, fact, and event Extraction: identification of associations among entities and other information in text.

Quantitative text analysis is a set of techniques stemming from the social sciences where either a human judge or a computer extracts semantic or grammatical relationships between words in order to find out the meaning or stylistic patterns of, usually, a casual personal text for the purpose of psychological profiling etc.

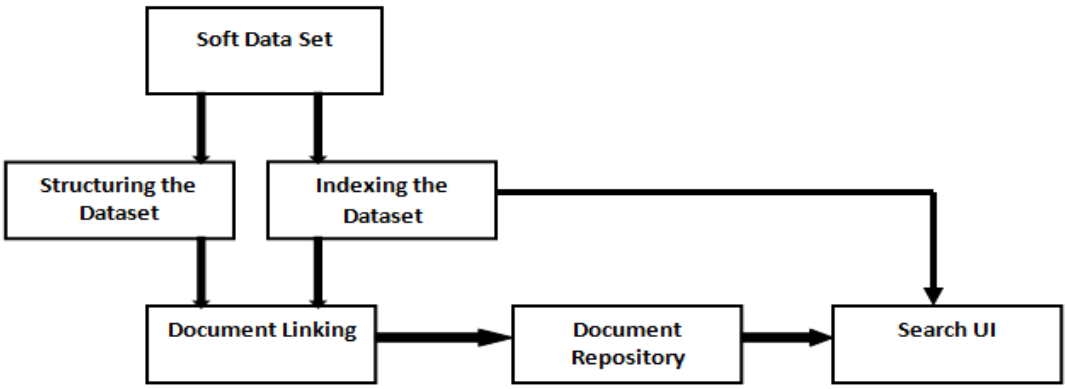

Figure 3.1: Overall System Process Diagram

The System Diagram components can be elaborated further: *Soft Dataset*: set of textual materials, on the Web or in a file system. *Structuring the Dataset*: require the structuring of document in file management system or, database, or content management system, for analysis.

Indexing the Dataset: All the structured documents are processed for indexing purpose so that document can be referred efficiently whenever required. *Document Linking*: Indexing of documents provide the ease of access to system which results in optimal document linking.

Document Repository: It maintains all the documents processed so far in interlinked state. Also documents can be appended in the repository and can be linked as well if required.

3.1 Data Flow Diagram for User Query

The flow of the data can be represented as follows:

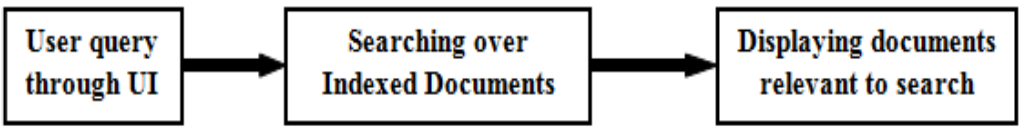

Figure 3.2: Data Flow Diagram

- Whenever any query is fired to the database using user interface, it searches over the complete database (also called as Data Repository).
- The indexed documents stored in database are scanned for given query.
- Thus, documents which are referred through searching can be produced.

3.2 Research Methodology

The main aim of the system is to design and develop document retrieval on the basis of user query and performs a pattern and regular expression based search. The very first step is to create a soft data set in searchable format. Following it, user search the data using a GUI, created. But it requires a well-structured dataset and also a indexed dataset. Here, we need extremely fast indexing rates and minimal indexing latency along with tight constraints of search response time in text retrieval systems.

3.2.1 Two-level Hierarchical Index Data Structure Design

The design for the index data structure considers efficient index generation while sustaining the search performance including search throughput and search response time. The key idea for indexing dataset is to remove overhead of linear search and its latency.

In this approach we keep a top-level hash-table called GHT (Global Hash Table) that maps unique terms in the document collection to a set of second-level hash tables. The second-level hash-table, called IHT (Document Interval Hash Table) is an index for a set (interval) of documents with contiguous Document IDs (documents numbered from 1 to D). Each term in IHT is mapped to a list of document IDs, where each document contains that term. For each such document the detailed occurrence positions called postings data is stored. This hierarchical data structure based approach gives two key advantages over single level hash table approach and sorted list of terms based approach as used in CLucene. First it avoids the need for repeated reorganization of data in the IHTs that get merged in the final merged index (GHT) as it only involves adding a reference to the IHT in GHT for terms contained in that IHT. Second, the terms in both IHT and GHT are organized in a hash-table structure.

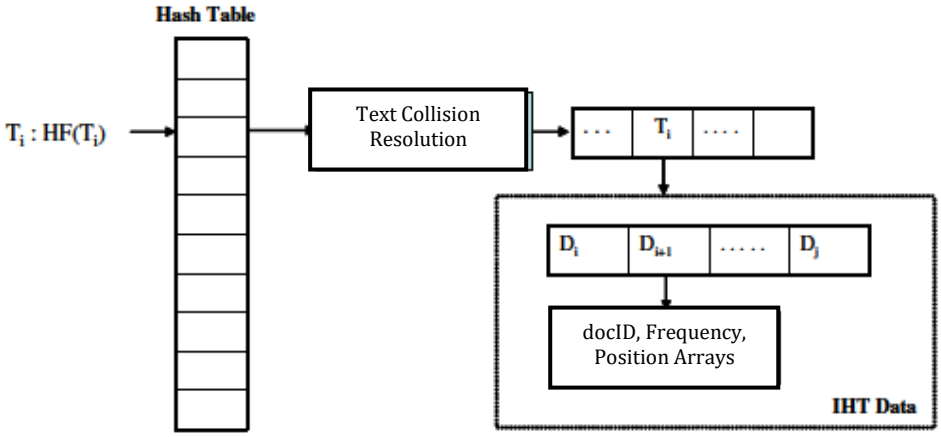

Figure 3.3: Interval Hash Table (IHT)

3.4 Algorithm

Our indexing algorithm has three main steps:

- Posting table (LHT) for each document is constructed without involving sorting of terms;
- Posting tables of k documents are merged into an IHT, which are then encoded appropriately;
- Encoded IHTs are merged into a single GHT in an efficient manner.

For IHT construction, first, the posting table (also referred to as LHT) for each of the documents is formed separately without involving sorting of terms. Then, each set of k posting tables (LHTs) are merged into one IHT per set.

From each LHT, each unique term is read one-by-one and inserted into the IHT, if it is a new term. Then, the document and postings data for this term is merged into the already available document and postings data for the term in the IHT. IHT helps in scalable distributed indexing by providing the ability to offload the construction of index for a set of documents with contiguous IDs to another processor before merging that into GHT.

- The GHT is constructed by merging IHTs one at a time into the GHT. The steps for merging an IHT into the GHT are as follows:Insert pointers to the IHT data, including encoded IHT data array and the positions array, into the Array-Of-IHTs. This insertion happens at that entry in Arrayof-IHTs which represents the document-interval corresponding to the current IHT being read. In Fig. 3, the entry g points to IHT(g). Also, store a base document ID for this IHT. The base ID is the sum of the base ID of the previous IHT and the number of documents in it (or the document interval size, if all IHTs contain same number of documents).
- The unique term list in the IHT is traversed. For each term, position of that term is identified in the GHT using hash-function evaluation and term collision resolution. Then, in the IHT-list for that term, the current IHT number is inserted. Fig. 3 illustrates how IHT(g), is merged into the GHT. First (Step-S1 in Fig. 3), IHT(g) is

pointed to by the appropriate location in the Array-of-IHTs. Then (Steps-S2(a) & S2(b) in Fig. 3), IHT(g), is inserted into both IHT-lists corresponding to the terms Ti and Tj in the GHT. The above merge process does not involve re-organizing of the IHT data while merging it into GHT, in contrast, to typical indexing approaches which re-organize the segment data when merging it into the final merged segment. This makes GHT/IHT design efficient for distributed indexing.

3.5 Result Analysis

We implemented our GHT/IHT based indexing data structures and algorithm, Pghtl, on the original CLucene codebase (v0.9.20). We also implemented CLucene based distributed indexing algorithm, Porgl, where we maintain the index in memory using the RAMDirectory. We studied the scalability of Porgl and Pghtl. The text data was extracted from document files and loaded equally into the memory of the producer nodes, before, the indexing time measurement is started. For Porgl, we used a value of k so that only one segment is created from all the text data fed to a Producer so as to get its best indexing throughput. For indexing latency measurement we had to however choose the same lower value of k = 256 as used for Pghtl, so that we could compare it meaningfully with Pghtl.

Indexing Throughput Analysis

In this section we study the scalability of indexing throughput with variation in number of processor nodes and input text data size.

- **Strong Scalability Study:** In the strong scalability experiment, the input data size for an index group remains constant while the size of the group is increased. We indexed large volumes of data.

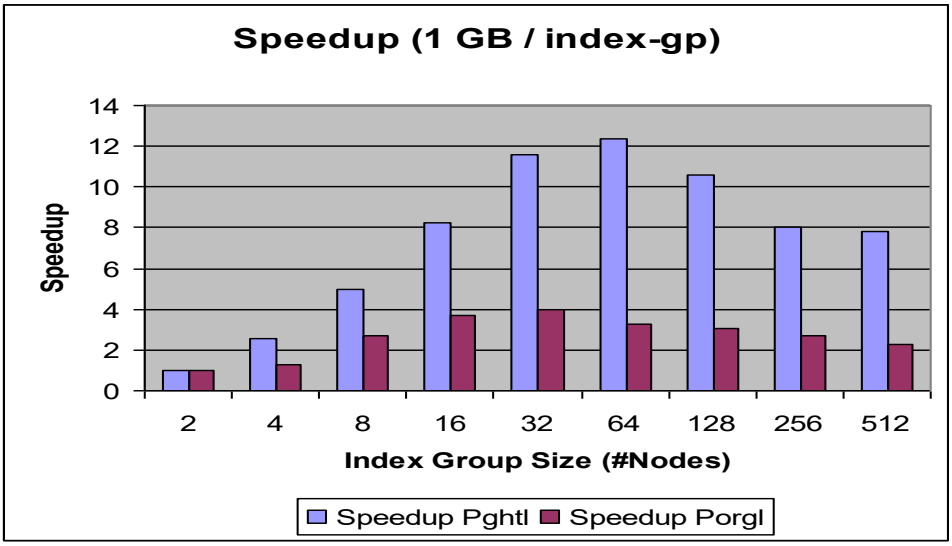

Figure 3.4: Speedup (strong scalability)

- **Weak Scalability:** In this experiment the data to be indexed per index group is increased from 50MB to 1.6 GB. We improve the weak scalability of distributed text indexing compared to CLucene by employing our more efficient algorithm Pghtl.

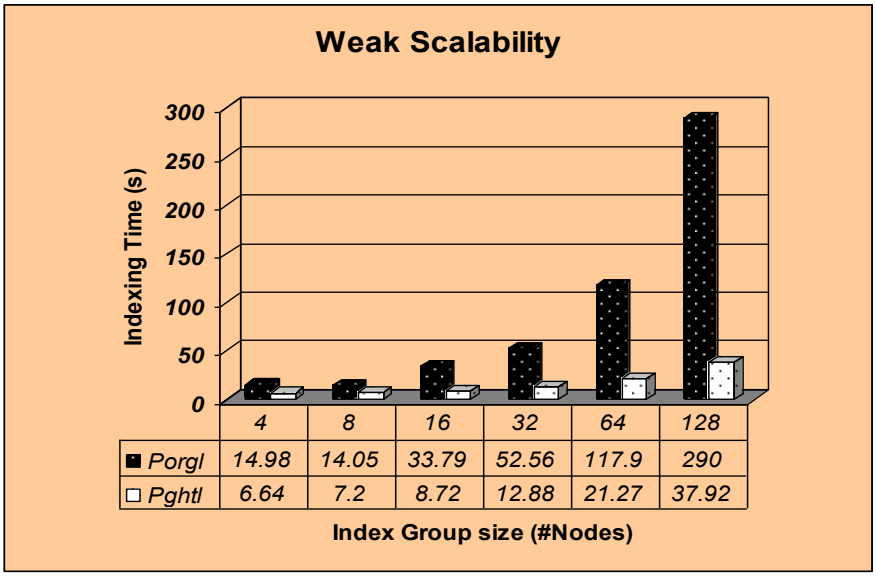

Figure 3.5: Weak Scalability Study

- **Indexing Latency Analysis** We measure the indexing latency as the time from the start of document indexing to the time the index for the document is available for search. Fig. 5.7 displays the variation in indexing latency with increase in size of the group, G. The amount of data indexed per group is maintained constant at 1 GB and the number of documents per IHT in Pghtl is kept as 256, and is the same as number of documents per first-level segment in Porgl. We can see that the indexing latency for Pghtl is one order of magnitude (around 9.91x) better than Porgl. The indexing latency is dependent on the segment merge time at the Consumer node in case of Porgl and since this is inefficient it leads to a larger latency and this increases with the increase in the number of segments to merge. In case of Pghtl the merge of IHT-meta-data is much more efficient and hence it is better.

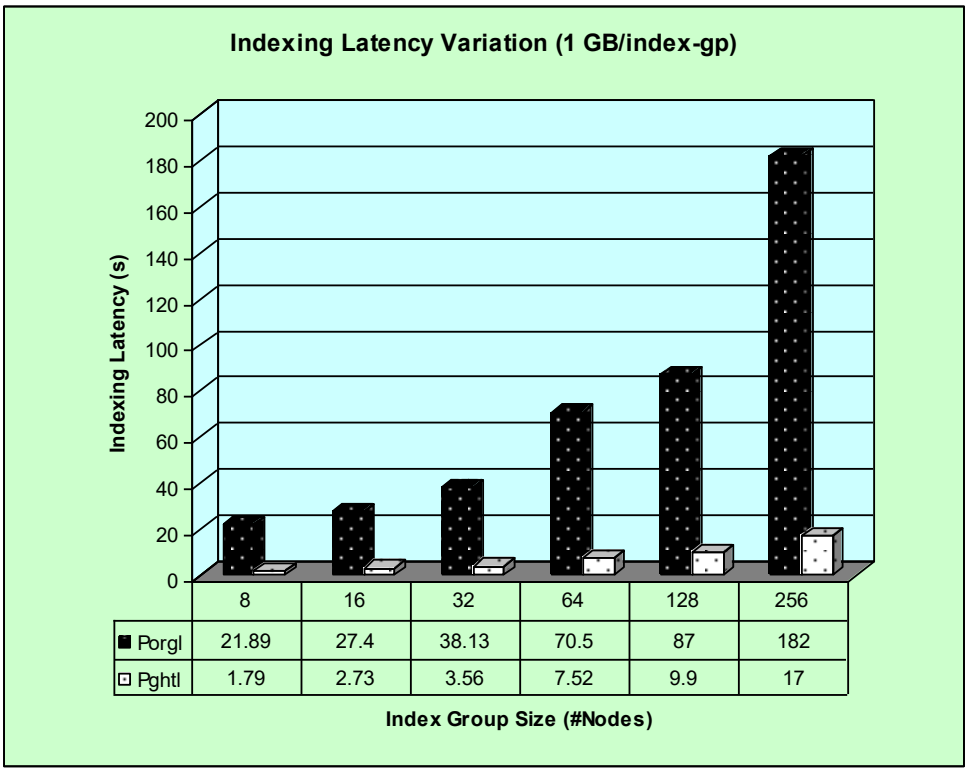

Figure 3.6: Indexing Latency Variation

4. CONCLUSION:

The main advantage of the document retrieval using text is, people who are not having any prior knowledge of Database systems, Database Management System can also access the information from database. Many of the times user do not have prior knowledge about the implementation and arrangement of database. Hence there is no need to have prior knowledge of the system. Other advantages are like there is no need to learn any query processing language for easy and efficient retrieval of Data. The Document Retrieval System will accept keyword or sentence as a query and gives output in form of list of documents. It is very much useful for the people who do not have any prior knowledge of database and terms feed in database. We have used modified like query to select search terms. With the help of this system we can easily access the database and retrieve the useful and relevant information whenever it is needed.

REFERENCES:

1. Debnath Bhattacharyya, Poulami Das," Unstructured Document Categorization: A Study", International Journal of Signal Processing, Image Processing and Pattern Recognition.
2. Weiguo Fan," Tapping into the Power of Text Mining", article accepted for publication at the Communications of ACM, February 16, 2005.
3. V.V.Jaya Rama Krishnaiah, D.V.Chandra Sekhar, Dr. K.Ramchand H Rao, Dr. R Satya Prasad," Predicting the Diabetes using Duo Mining Approach", published in International Journal of Advanced Research in Computer and Communication Engineering Vol. 1, Issue 6, August 2012.
4. K.Sreerama Murthy, Dr G. Samuel Varaprasad Raju, Dr C.Sunil Kumar," Text Mining For Retrieving The Vital Information", published in International Journal of Research in Computer and Communication Technology, Vol 3, Issue 1, January- 2014
5. Manish Sharma, Rahul Patel," A Survey on Information Retrieval Models, Techniques And Applications", published in International Journal of Emerging Technology and Advanced Engineering, November 2013.

6. B.Ganga," Phrase Based Document Retrieving by Combining Suffix Tree index data structure and Boyer-Moore faster string searching algorithm", published in International Journal of Advancements in Research & Technology, Volume 3, Issue 3, March-2014.
7. Ian H. Witten, Computer Science, University of Waikato, Hamilton, New Zealand. Reference book referred" Text mining".
8. Roi Blanco González," Index Compression for Information Retrieval Systems", Ph.D. THESIS, University of A Coruña, 2008.
9. Deepak Agnihotri, Kesari Verma, Priyanka Tripathi," Pattern and Cluster Mining on Text Data", published in Fourth International Conference on Communication Systems and Network Technologies,2014.
10. R. Sagayam, S.Srinivasan, S. Roshni," A Survey of Text Mining: Retrieval, Extraction and Indexing Techniques",published in International Journal Of Computational Engineering Research Vol. 2 Issue. 5,September 2012.
11. Sonali Vijay Gaikwad, Prof Archana Chaugule,Swapnil Kulkarni," Performance Comparison For Text Mining Methods: Review"published in Int. J. Adv. Engg. Res. Studies/IV/I/Oct.-Dec,2014/01-04.
12. Ning Zhong, Yuefeng Li, and Sheng-Tang Wu," Effective Pattern Discovery for Text Mining" published in IEEE TRANSACTIONS On Knowledge And Data Engineering, Vol. 24, No. 1, January 2012.
13. Bhushan Inje, Ujawla Patil," Operational Pattern Revealing Technique in Text Mining",published in 2014 IEEE Students' Conference on Electrical, Electronics and Computer Science.
14. Ziqi Wang, Gu Xu, Hang Li, and Ming Zhang," A Probabilistic Approach to String Transformation",published in IEEE Transactions On Knowledge And Data Engineering, Vol. 26, No. 5, May 2014.
15. Saima Hasib, Mahak Motwani, Amit Saxena," Importance of Aho-Corasick String Matching Algorithm in Real World Applications" published in International Journal of Computer Science and Information Technologies, Vol. 4 (3) , 2013, 467-469.