International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

Automated Query Generation Of Relational DBMS For Information And Knowledge Extraction

[1]Prof. Vaishali Londhe , [2]Prof. Nilima Nikam, [3]Miss. Surekha Naik

Dept. of Computer Engineering, University of Mumbai, India

ABSTRACT

An Information extraction systems traditionally implemented as a pipeline of special-purpose processing modules targeting the extraction of a particular kind of information. A major drawback is, whenever a new extraction goal emerges or a module is improved, extraction has to be reapplied from scratch to the entire text corpus even though a small part of corpus might be affected. By using database queries, information extraction enables the generic extraction and minimizes re-processing of data. Furthermore, this provides automated query generation components so that, casual users no need to learn the query language in order to perform extraction. To demonstrate the feasibility of our incremental extraction approach, experiments can be performed to highlight two important aspects of an information extraction system: efficiency and quality of extraction results. The existing extraction frameworks do not provide the capabilities of managing intermediate processed data such as parse trees and information. It is also extended to per-sentence extraction, it is important to notice that the query language itself is capable of defining patterns across multiple sentences. Hence, in order to provide nearest and good result the incremental approach is compared with the existing systems.

Key words: information storage and information retrieval, Text Mining, query languages.

I. INTRODUCTION

Improving the ability of computer systems to process text is a significant research Challenge. IE has been an active research area that seeks techniques to uncover information from a large collection of text. Examples of common IE tasks include the identification of entities (such as protein names), extraction of relationships between entities (such as interactions between a pair of proteins) and extraction of entity attributes (such as co reference resolution that identifies variants of mentions corresponding to the same entity) from text. Many applications are based on partially structured databases, where structured data conforming to a schema is combined with free text. Information extraction systems traditionally implemented as a pipeline of special-purpose processing modules targeting the extraction of a particular kind of information. Most recent IE approaches are suitable for only static corpora.

1.1 Natural Language Processing

IBM Research has over 200 people working on Unstructured Information Management technologies with a strong focus on Natural Language Processing (NLP). These researchers are engaged in activities ranging from natural language dialog, information retrieval, topic-tracking, named-entity detection, document classification

and machine translation to bioinformatics and open-domain question answering. An analysis of these activities strongly suggested that improving the organization's ability to quickly discover each other's results and rapidly combines different technologies and approaches would accelerate scientific advance.

1.2 Objectives and scope

Information Extraction is a technology that is futuristic from the user's point of view in the current information-driven world. Rather than indicating which documents need to be read by a user, it extracts pieces of information that are salient to the user's needs.

Document search and retrieval applications tailored to the needs of life science. Structure extraction is useful in a diverse set of applications Similarly, news archives contain information useful to analysts tracking financial transactions, or to government agencies that monitor infectious disease outbreaks. etc.

II. LITERATURE SURVEY

2.1 Existing system

Information extraction has been an active research area over the years. The main focus has been on improving the accuracy of the extraction systems, and IE has been seen as a one-time execution process. Such paradigm is inadequate for real-world applications when IE is seen as long as running processes. An example of a real-world application of IE is the extraction from evolving text, such as the frequent update of the content of web documents. [3]

A wealth of information is hidden within unstructured text. This information is often best exploited in structured Current information extraction techniques extract relations from a text database by examining every document in the database, or use filters to select promising documents for extraction.

2.2 Rule-based IE Approaches

In general, a document is broken up into chunks (e.g., sentences or paragraphs), and rules or patterns applied to identify entities. Different operations such as joins in RDBMS are performed over extracted facts that stored in various database tables. Rules are then applied to integrate different types of extracted facts. However, these rules are not capable of querying parse trees.

2.3 Machine learning approaches for IE

Machine learning a branch of artificial intelligence, concerns the construction and study of systems that can learn from data. For example, a machine learning system could be trained on email messages to learn to distinguish between spam and non-spam messages. After learning, it can then be used to classify new email messages into spam and non-spam folders.

III. MOTIVATION & PROBLEM DEFINITION

A typical IE setting involves a pipeline of text processing modules in order to perform relationship extraction.

- These include:
  - Sentence splitting: identifies sentences from a paragraph of text.
  - Tokenization: Identifies word tokens such as nouns, verbs, adverbs from sentences.

- Named entity recognition: Identifies mentions of entity types of interest. Entity type can be person name, location etc. Classify word token into predefined categories.
- Syntactic parsing: Identifies grammatical structure of sentences.
- Pattern matching: Obtain relationship based on a set of extraction pattern that utilize lexical, syntactic and semantic features.

Disadvantages of previous existing system

- Do not provide capabilities for managing intermediate processed data.Lead to the need for reprocessing of entire text collection which can be computationally expensive.

3.1Proposed IE

In our proposed IE we try to reduce this computational intensity. In this extraction framework intermediate output of each text processing component is stored so that only the improved component has to be deployed to the entire corpus. Extraction is then performed on both previously processed data from unchanged component as well as updated data generated by the improved component. Performing such kind of incremental extraction can result in tremendous reduction in processing time.

Proposed information extraction is composed of two phases:

- Initial phase:
  - Perform one time parse
  - Entity recognition (classify each word token into a predefined category or entity type) and tagging (each entity type is named with a tag name).
  - Generated syntactic parse trees and semantic entity tagging of the processed text is stored in a parse tree database (PTDB).
- Extraction Phase:
  - By issuing database queries to PTDB extraction is achieved.
  - To express extraction pattern we designed and implemented parse tree query language called parse tree query language (PTQL).
  - Writing PTQL queries requires much user effort so we propose a system that will automatically generate PTQL queries based on user's keyword queries.

3.2 Problem Definition

Our approach provides Text mining involves application of techniques from areas such as information retrieval, natural language processing, information extraction and data mining. These various stages of a text-mining process are combined together into a single workflow. As automated query generation, so that casual users do not have to learn the query language in order to perform extraction. we performed experiments to highlight two important aspects of an information extraction system: efficiency and quality of extraction results. For proposed work, we will extend the support of Stanford parser, so that they stored Data in PTDB and queried using PTQL. We will expand the capabilities of PTQL, such as the support of regular expression and the utilization of redundancy to compute confidence of the extracted information. This approach use for Biomedical Applications as well as Research work Applications.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

IV. SYSTEM FRAMEWORK

4.1. Overview

Text mining involves the application of techniques from areas such as information retrieval, natural language processing, information extraction and data mining. These various stages of a text-mining process can be combined together into a single workflow.

4.1.1 Information Retrieval (IR)

Systems identify the documents in a collection which match a user's query. IR systems are often used in libraries, where the documents are typically not the books themselves but digital records containing information about the books.IR systems allow us to narrow down the set of documents that are relevant to a particular problem. As text mining involves applying very computationally-intensive algorithms to large document collections, IR can speed up the analysis considerably by reducing the number of documents for analysis.

4.1.2 Natural Language Processing (NLP)

NLP can perform some types of analysis with a high degree of success. Shallow parsers identify only the main grammatical elements in a sentence, such as noun phrases and verb phrases, whereas deep parsers generate a complete representation of the grammatical structure of a sentence

- **4.3Information Extraction (IE)**

It is the process of automatically obtaining structured data from an unstructured natural language document.IE systems rely heavily on the data generated by NLP systems.

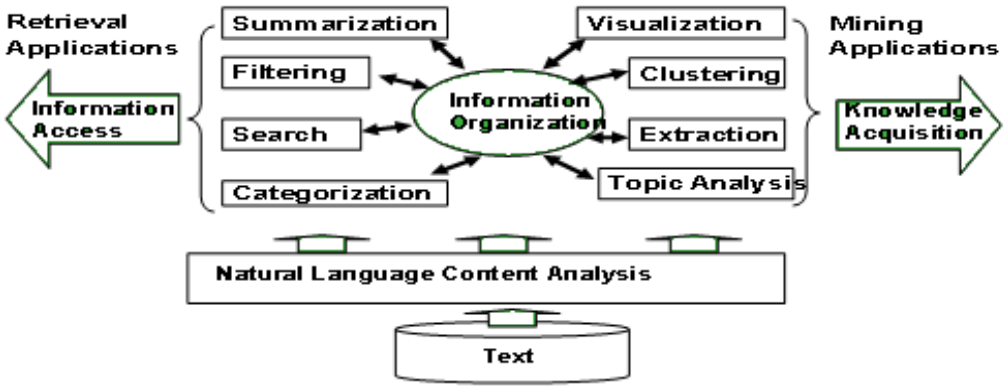


Fig 4.1: Conceptual Framework of Text Information System

Fig 4.1 shows the Conceptual Framework of Text Information System consist the Retrieval applications & Mining applications. Natural language content analysis access text and send it to information organization, which perform operations like Topic analysis, Extraction, Clustering, Visualization, Summarization, Filtering, Search, Categorization etc. Information organization provide information access & knowledge acquisition.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

V. METHODOLOGY

5.1 System Approach

A Novel Database-Centric Framework for Information Extraction. We first give an overview of our approach, and discuss each of the major components of our system. Abstract— Information extraction systems traditionally implemented as a pipeline of special-purpose processing modules targeting the extraction of a particular kind of information. A major drawback is, whenever a new extraction goal emerges or a module is improved, extraction has to be reapplied from scratch to the entire text corpus even though a small part of corpus might be affected. By using database queries, information extraction enables the generic extraction and minimizes re-processing of data. Furthermore, this provides automated query generation components so that, casual users no need to learn the query language in order to perform extraction can be performed to highlight two important aspects of an information extraction system: efficiency and quality of extraction results. The existing extraction frameworks do not provide the capabilities of managing intermediate processed.

Our approach composed basic two phases:

- **Initial phase:** for processing of text and
- **Extraction phase:** for using database queries to perform extraction.

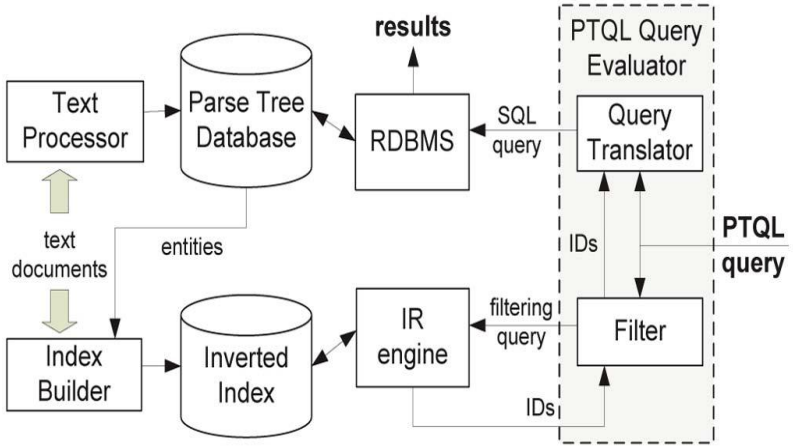


Fig4.2 System architecture of the PTQL framework

As shown in fig4.2 the Text Processor in the initial phase is responsible for corpus processing and storage of the processed information in the Parse Tree Database (PTDB). The extraction patterns over parse trees can be expressed in our proposed parse tree query language. In the extraction phase, the PTQL query evaluator takes a PTQL query and transforms it into keyword-based queries and SQL queries, which are evaluated by the underlying RDBMS and information retrieval (IR) engine. To speed up query evaluation, the index builder creates an inverted index for the indexing of sentences according to words and the corresponding entity types. Fig 5.1 illustrates the system architecture of our approach.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

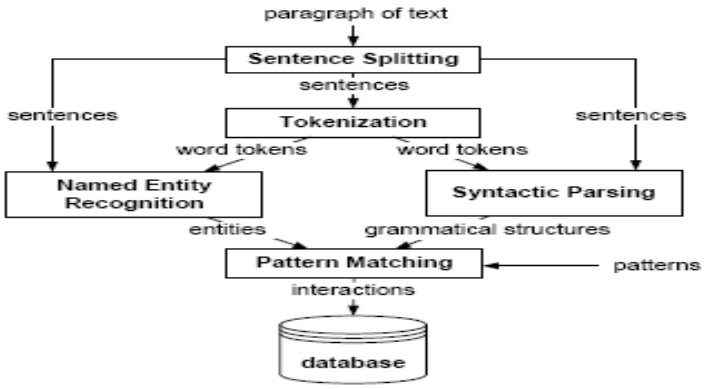


Fig.A workflow of text processing modules that takes a paragraph of text as input to perform relationship extraction

A typical IE setting involves a pipeline of text processing modules in order to perform relationship extraction. These include:

1. Sentence splitting: identifies sentences from a paragraph of text.

2. Tokenization: identifies word tokens from sentences.

3. Named entity recognition: identifies mentions of entity types of interest.

4. Syntactic parsing: identifies grammatical structures of sentences.

5. Pattern matching: obtains relationships based on a set of extraction patterns that utilize lexical, syntactic, and semantic features.

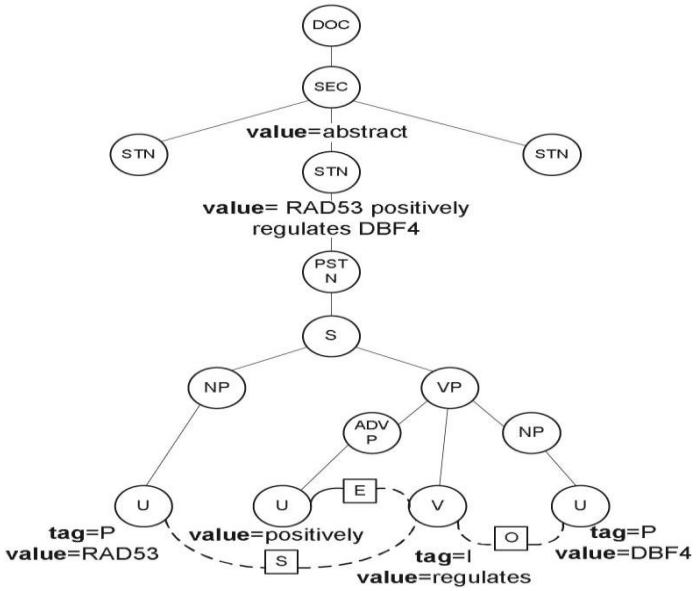


Fig5.4: An example of a parse tree for a document, which includes sections of the document,

The linkage contains three different links: the S link connects the subject-noun RAD53 to the transitive verb regulates, the O link connects the transitive verb regulates to the direct object DBF4 and the E link connects the

verb-modifying adverb positively to the verb regulates. The square box on a dotted line indicates the link type between two words. Each l ----------

node in a parse tree has value and tag attributes. The value attribute stores the text representation of a node, while the tag attribute indicates the entity type of a leaf node. For instance, a protein is marked with a tag P, a drug name with a tag D, and an interaction word is marked with I.

VI. SYSTEM DESIGN

6.1 Flowchart of System

It shows the flow of actual system activities perform during operation. Fig 6.1 shows flow of system.

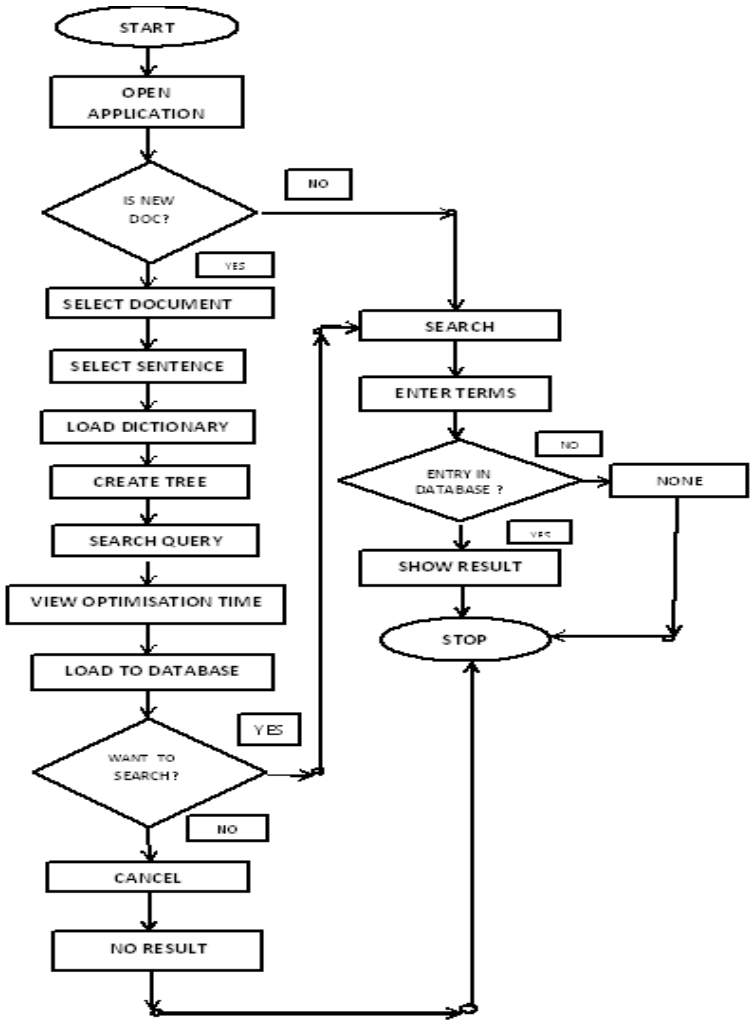


Fig 6.1: Flow of System

6.2 Phases of System

Proposed information extraction is composed of two phases:

- **Initial phase**:

➢ Perform one time parse

- Entity recognition (classify each word token into a predefined category or entity type) and tagging (each entity type is named with a tag name).
- Generated syntactic parse trees and semantic entity tagging of the processed text is stored in a parse tree database (PTDB).

- **Extraction Phase:**

- By issuing database queries to PTDB extraction is achieved.
- To express extraction pattern we designed and implemented parse tree query language called parse tree query language (PTQL).
- Writing PTQL queries requires much user effort so we propose algorithms that will automatically generate PTQL queries based on user's keyword queries.

Text Processor performs initial phase and stores processed information in the PTDB. Index Builder:

- Recognize entities and replace entities with entity type or identifier.
- It will index each sentence.
- Store the result in inverted index which include original sentence and sentence with entities replaced by identifiers.

Extraction pattern can be expressed in parse tree query language (PTQL).To evaluate PTQL queries on PTDB: The Query Translator generates SQL queries from PTQL queries. Optimizations for each PTQL query: The Filter module first generates an keyword-based query to efficiently prune irrelevant sentences. The Query Translator generates a SQL query equivalent to the PTQL query. It performs the actual extraction only on relevant sentences.

VII. RESULTS

We can get progressive corpus without extraction from scratch by storing the intermediate results in the table and retrieve the document by using PTQL query. We first illustrate the performance of our approach in terms of query evaluation and the time savings achieved through incremental extraction. We evaluate the extraction performance for our two approaches

1. Automatic training set driven query generation & data processing i.e. Without PTQL.
2. Automatic Query generation by using user keyword based query generation & data processing i.e. With PTQL.

7.1 Input Data Sample1: Medical Data

Your tonsils and adenoids are part of your lymphatic system Treatments include surgery, radiation therapy, photodynamic therapy (PDT), and biologic therapy.

7.1.1 Document parse tree:

When we select document & sentence then contents of sentence are arrange in standard tree structure to represent the unstructured data. A constituent tree is a syntactic tree of a sentence with the nodes represented by part-of-speech tags and leafs corresponding to words in the sentence. Fig 8.1 shows Snapshot of Document parse tree without PTQL.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

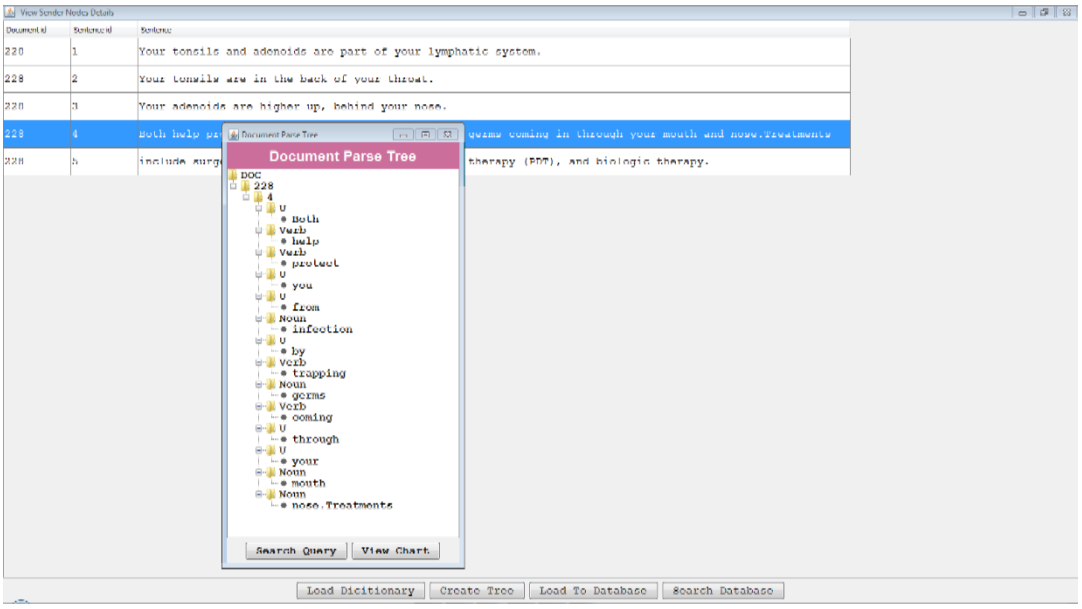


Fig 8.1: Snapshot of Document parse tree without PTQL

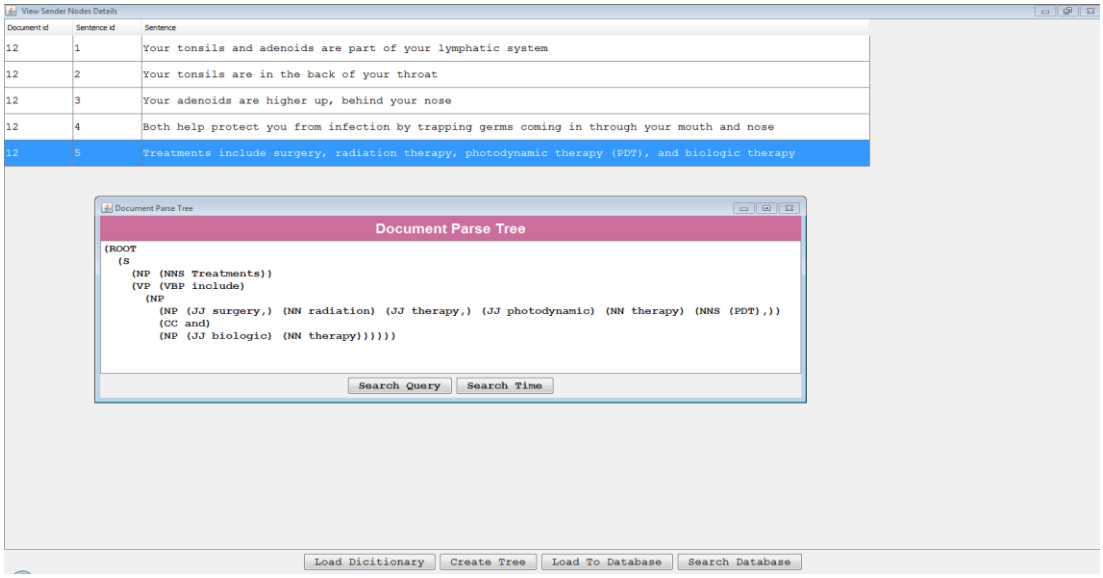


Fig8.2: Snapshot of Document parse tree with PTQL

7.1.2 Search query:

We showed that we can retrieve the document file names based on the user query and the query searches based on the phrase based search which retrieves semantic data. We can get progressive corpus without extraction from scratch by storing the intermediate results in the table and retrieve the document by using PTQL query. Fig 8.3 shows Snapshot of Search Query without PTQL. Fig8.4 shows Snapshot of Search Query with PTQL

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

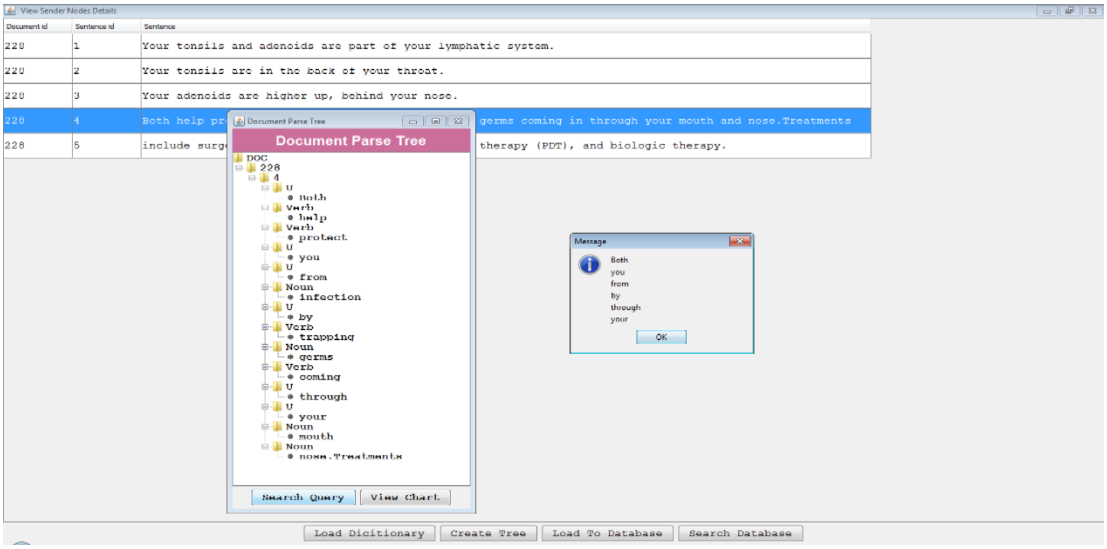


Fig8.3: Snapshot of Search Query without PTQL.

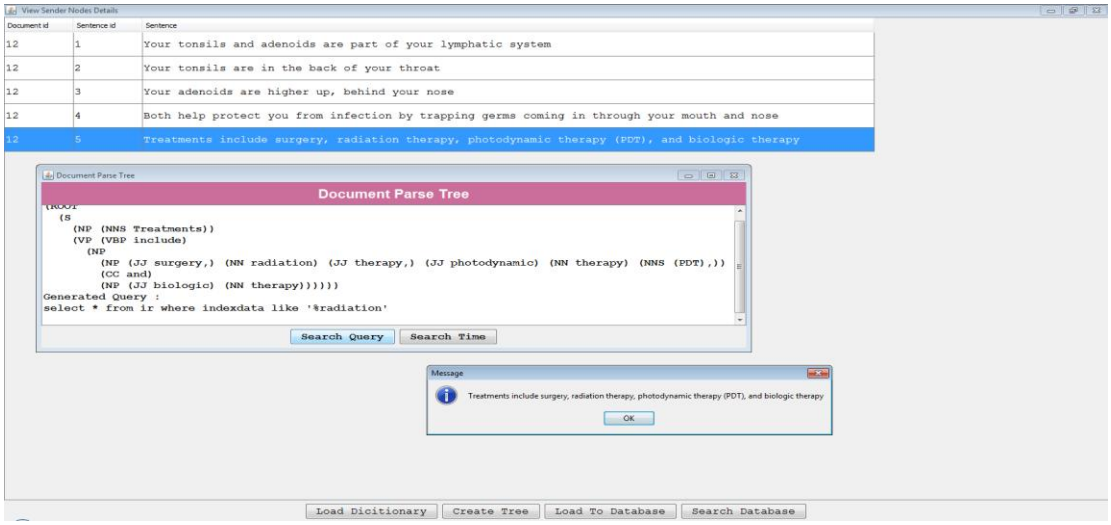


Fig8.4: Snapshot of Search Query with PTQL.

7.1.2.1. Time performance chart:

We are calculating the processing time for information in the database, these time changes can be represented in a chart format Fig 8.5 shows Snapshot of Time performance of search query without PTQL. And Fig 8.6 shows Snapshot of Time performance of search query with PTQL. Time performance in Micro seconds for search with PTQL evaluation & in Milliseconds for search without PTQL. The time performance indicates that our proposed framework is acceptable for real-time IE.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

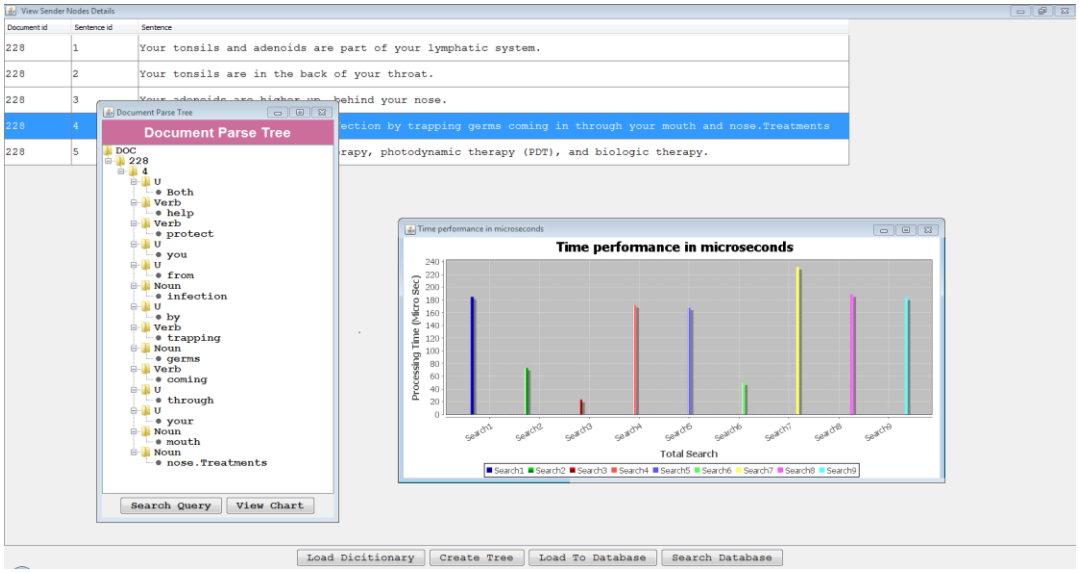
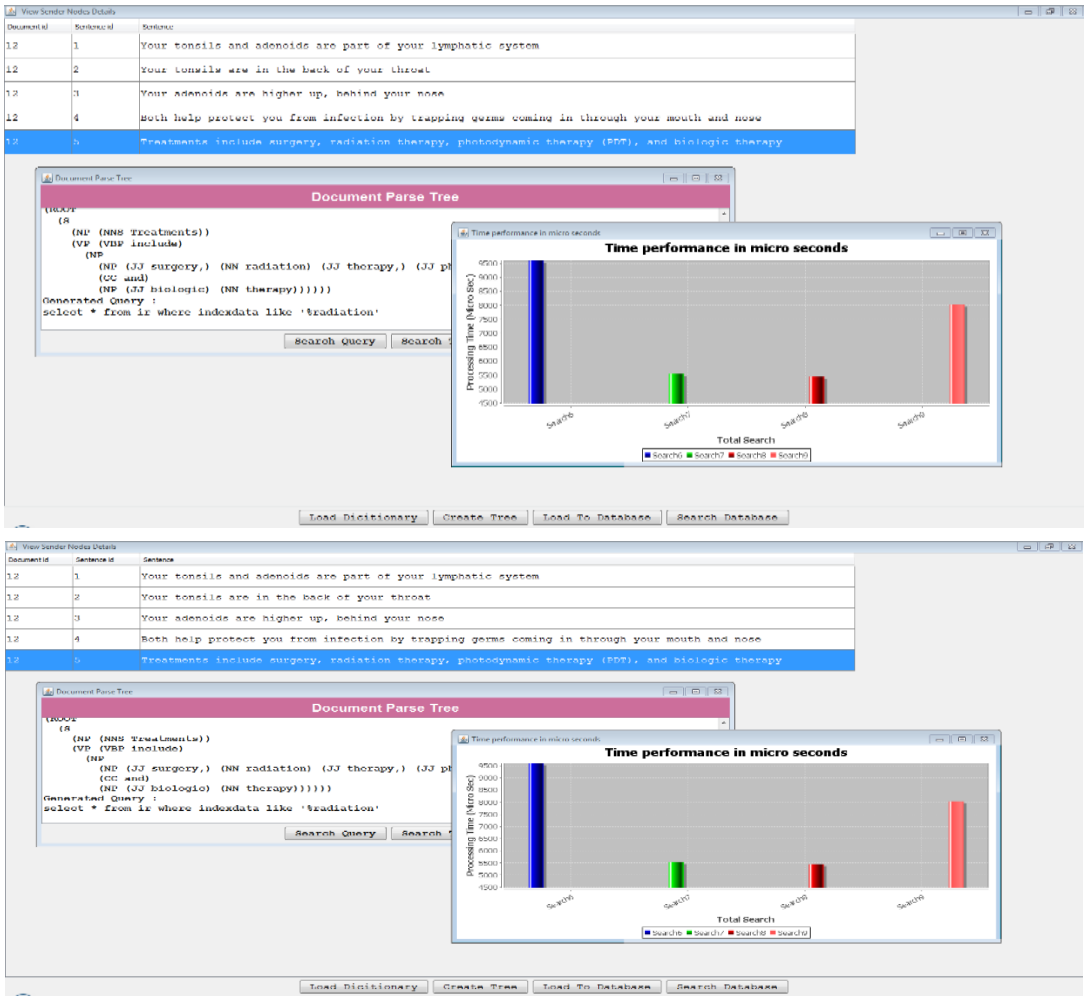


Fig 8.5: Snapshot of Time performance of search query without PTQL



. **Fig 8.6: Snapshot of Time performance of search query with PTQL.**

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

7.2 Input Data Sample 2: Biomedical Research Data.

Rad53 is a protein kinase required for cell-cycle arrest in response to DNA damage. Rad53 controls the S-phase checkpoint as well as G1 and G2 DNA damage checkpoints. It prevents entry into anaphase and mitotic exit.

7.2.1. Document parse tree:

When we select document & sentence then contents of sentence are arrange in standard tree structure to represent the unstructured data. Fig 8.7 shows Snapshot of Document parse tree without PTQL. Fig 8.8 shows Snapshot of Document parse tree with PTQL.

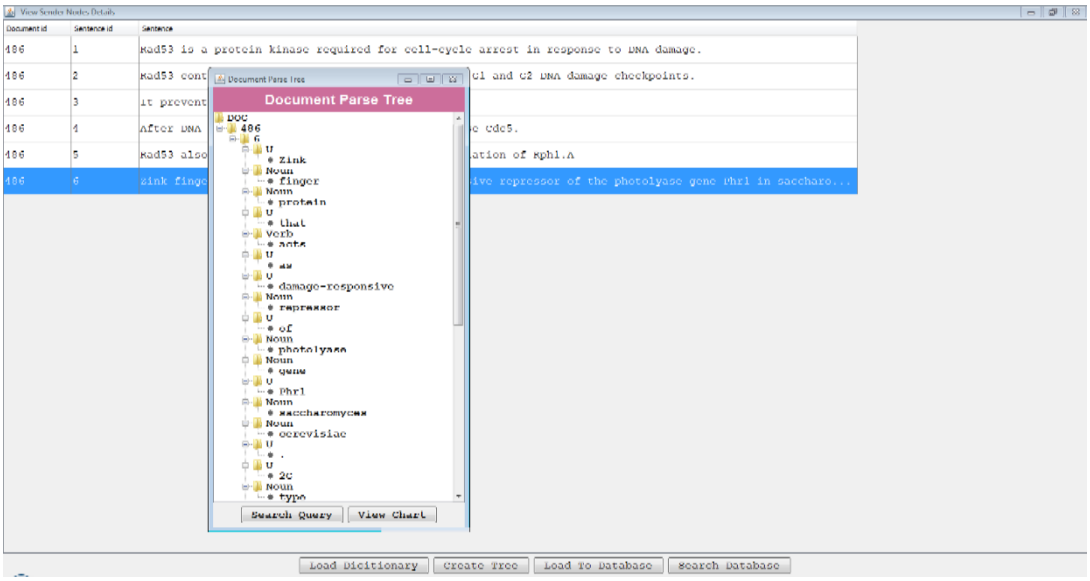


Fig 8.7: Snapshot of Document parse tree without PTQL.

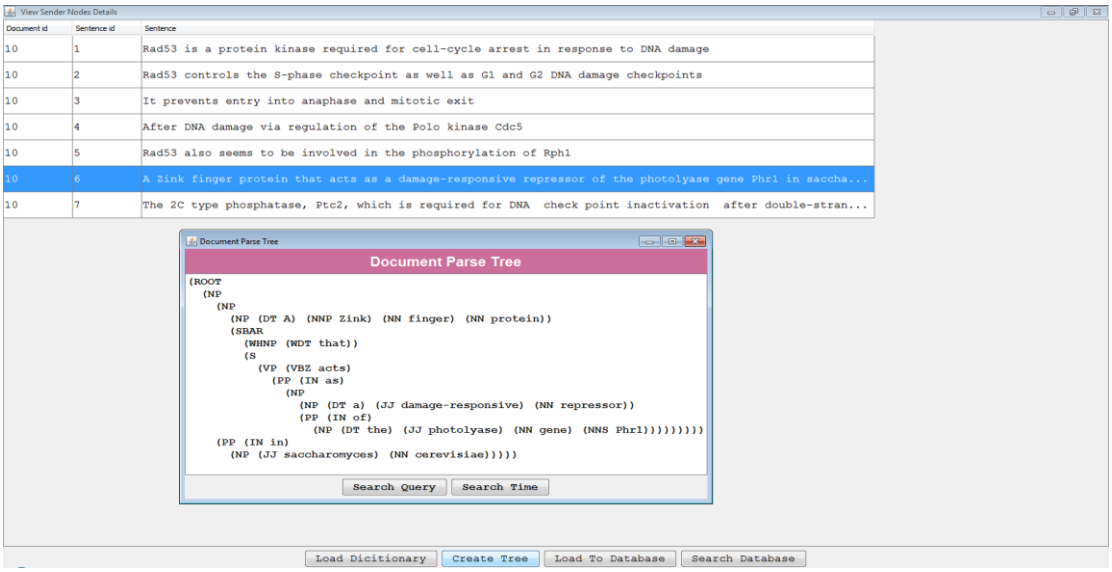


Fig8.8: Snapshot of Document parse tree with PTQL.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

7.2.2. Search query:

We showed that we can retrieve the document file names based on the user query and the query searches based on the phrase based search which retrieves semantic data. We can get progressive corpus without extraction from scratch by storing the intermediate results in the table and retrieve the document by using PTQL query. Fig 8.9 shows Snapshot of Search Query without PTQL. Fig 8.10 shows Snapshot of Search Query with PTQL.

Fig8.9: Snapshot of Search Query without PTQL.

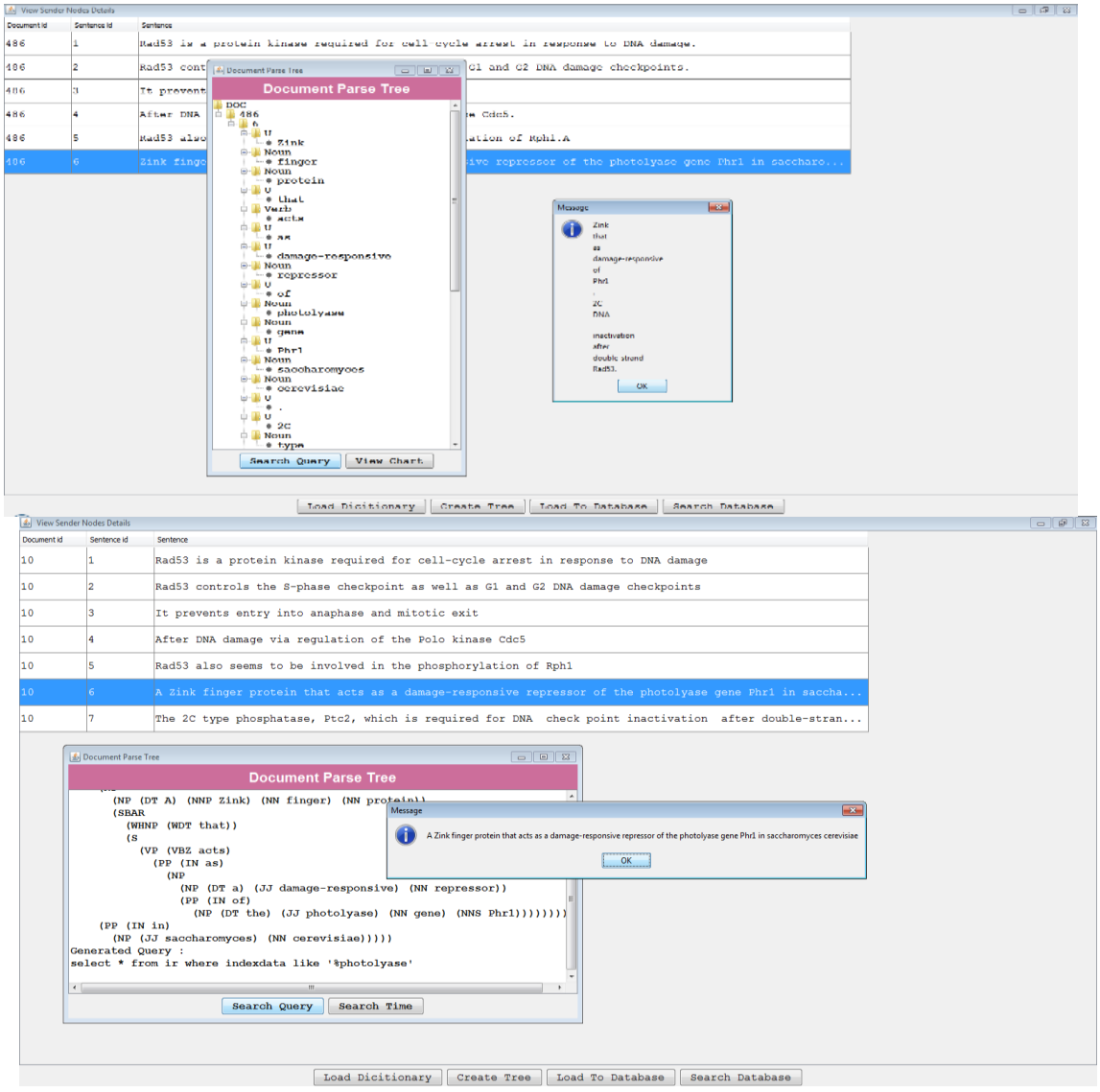


Fig 8.10: Snapshot of Search Query with PTQL.

7.2.3. Time performance chart:

We are calculating the processing time for information in the database, these time changes can be represented in a chart format Fig 8.11 shows Snapshot of Time performance of search query without PTQL.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

And Fig 8.12 shows Snapshot of Time performance of search query with PTQL. Time performance in Micro seconds for search with PTQL evaluation & in Milliseconds for search without PTQL. The time performance indicates that our proposed framework is acceptable for real-time IE.

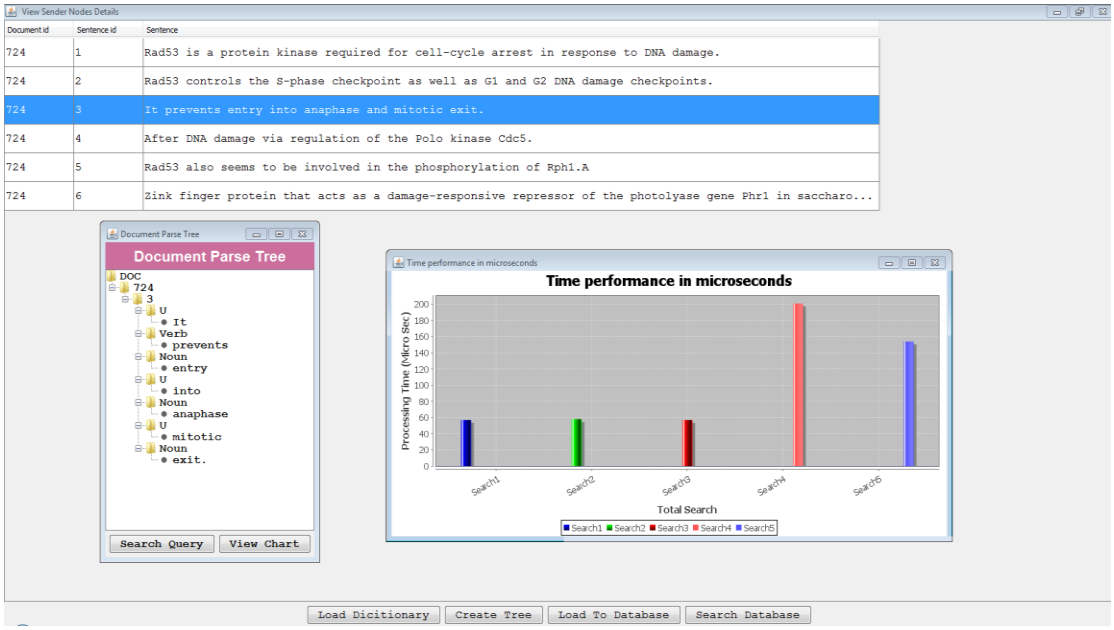


Fig 8.11: Snapshot of Time performance of search query without PTQL

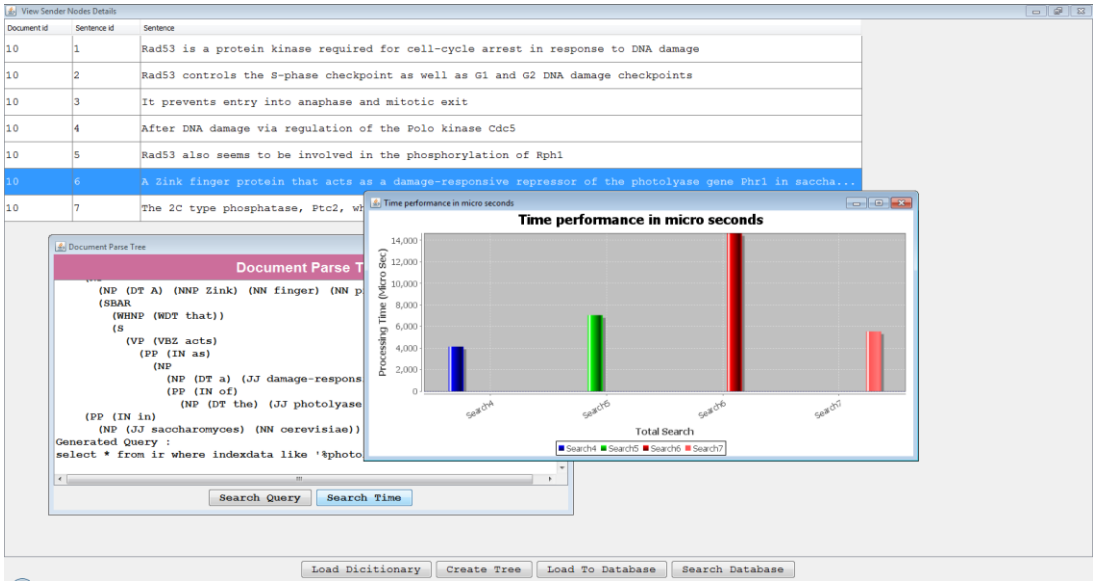


Fig 8.12: Snapshot of Time performance of search query with PTQL.

7.3 Input Data Sample3: Biomedical Data

RAD53, which activates DNA damage, positively regulates the DBF4 protein. Triazolam is metabolized by CYP3A4. Crossing is metabolized by CYP3A4. Rad: a unit of absorbed dose of ionizing radiation equal to an

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

energy of 100 ergs per gram of irradiated material. Control of the DNA Damage Checkpoint by Chk1 and Rad53 Protein Kinases through Distinct Mechanism.

7.3.1Document Parse Tree

When we select document & sentence then contents of sentence are arrange in standard tree structure to represent the unstructured data. Fig 8.13 shows Snapshot of Document parse tree without PTQL. Fig 8.14 shows Snapshot of Document parse tree with PTQL.

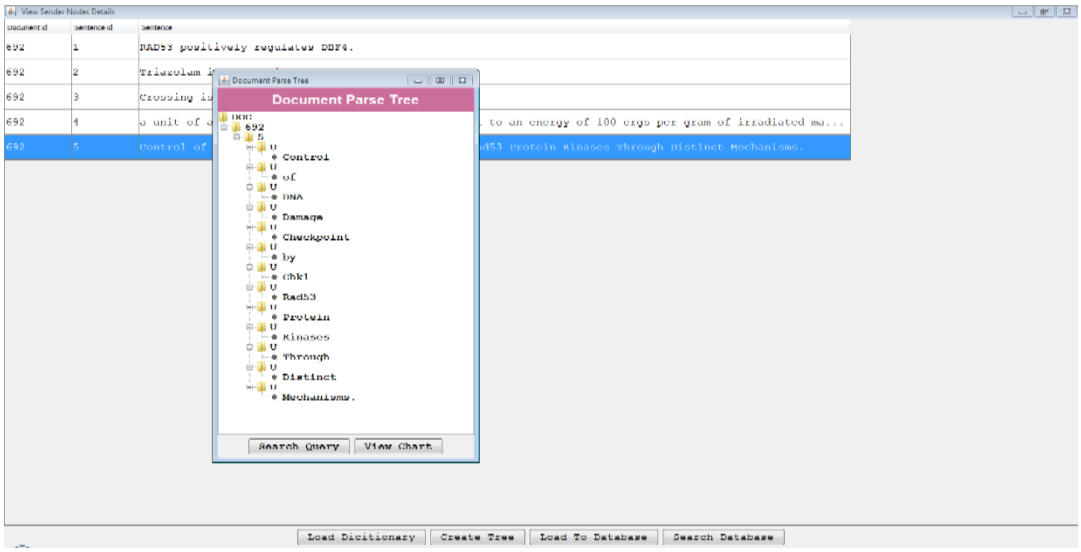


Fig 8.13.: Snapshot of Document Parse Tree without PTQL

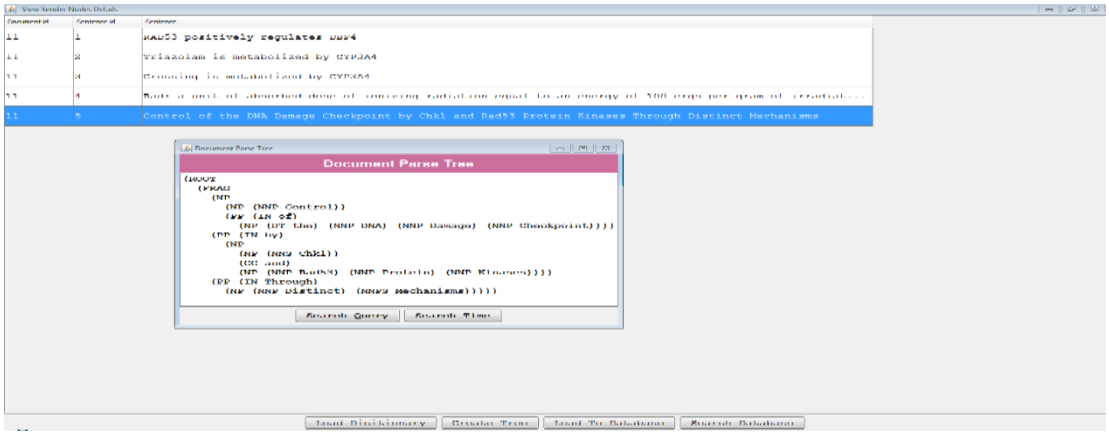


Fig 8.14: Snapshot of Document Parse Tree with PTQL

7.3.2. Search Query:

We showed that we can retrieve the document file names based on the user query and the query searches based on the phrase based search which retrieves semantic data. Fig 8.15 shows Snapshot of Search Query without PTQL. Fig 8.16 shows Snapshot of Search Query with PTQL.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

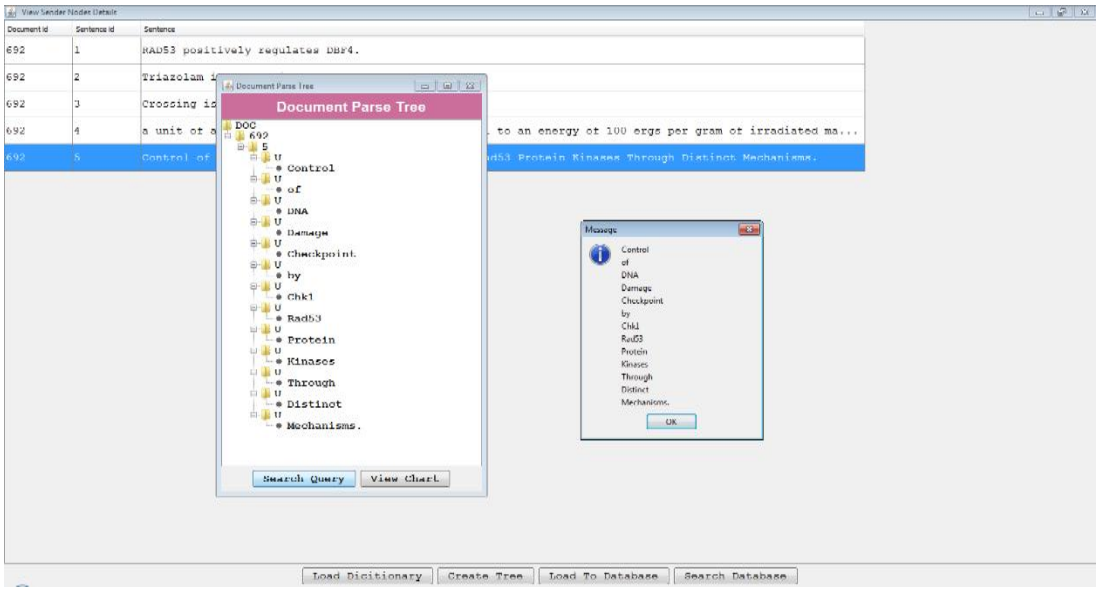


Fig 8.15: Snapshot of Search Query without PTQL

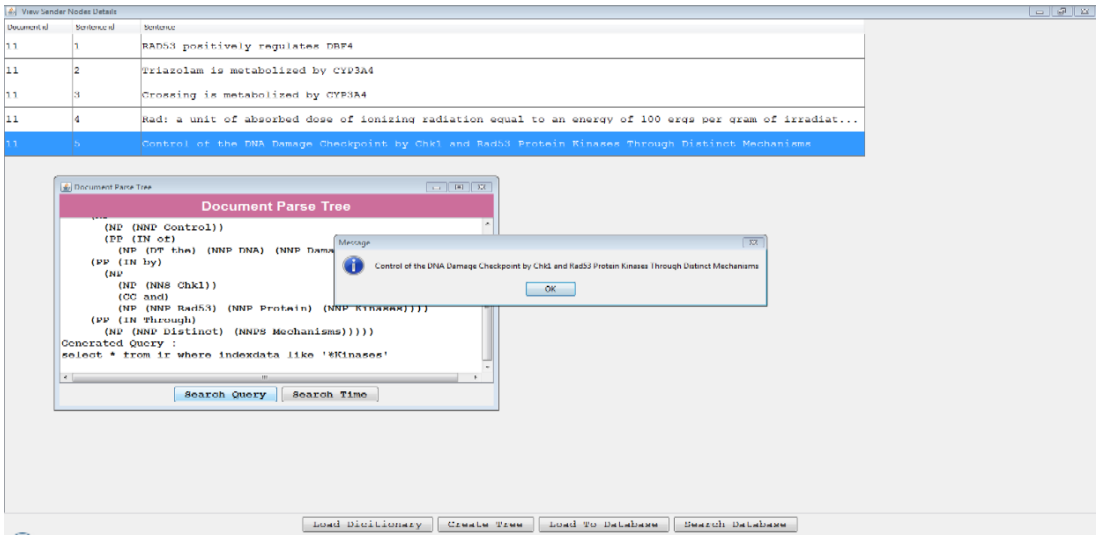


Fig 8.16: Snapshot of Search Query with PTQL

7.3.3. Time Performance

We are calculating the processing time for information in the database, these time changes can be represented in a chart format Fig 8.17 shows Snapshot of Time performance of search query without PTQL. And Fig 8.18 shows Snapshot of Time performance of search query with PTQL. Time performance in Micro seconds for search with PTQL evaluation & in Milliseconds for search without PTQL. The time performance indicates that our proposed framework is acceptable for real-time IE.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 11, November 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

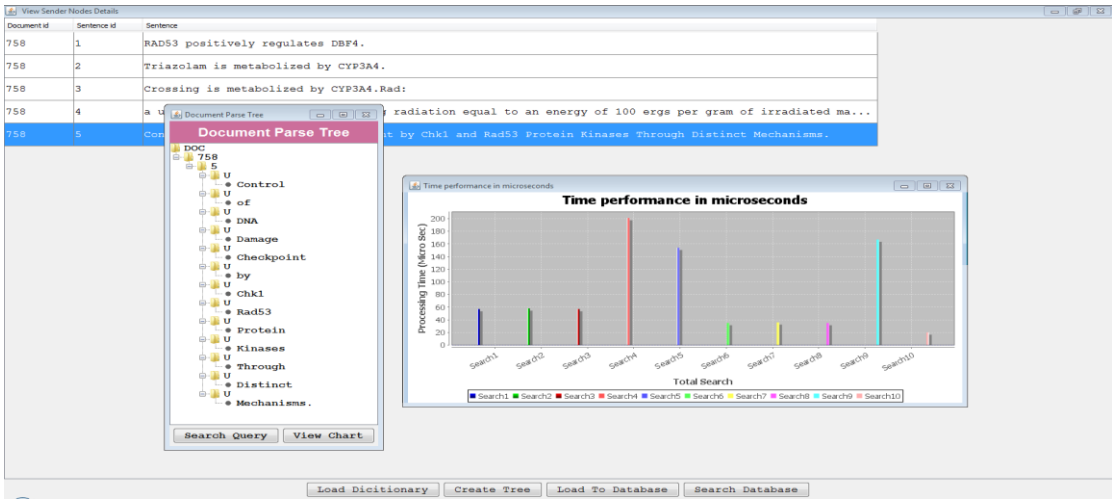


Fig 8.17: Snapshot of Time performance without PTQL.

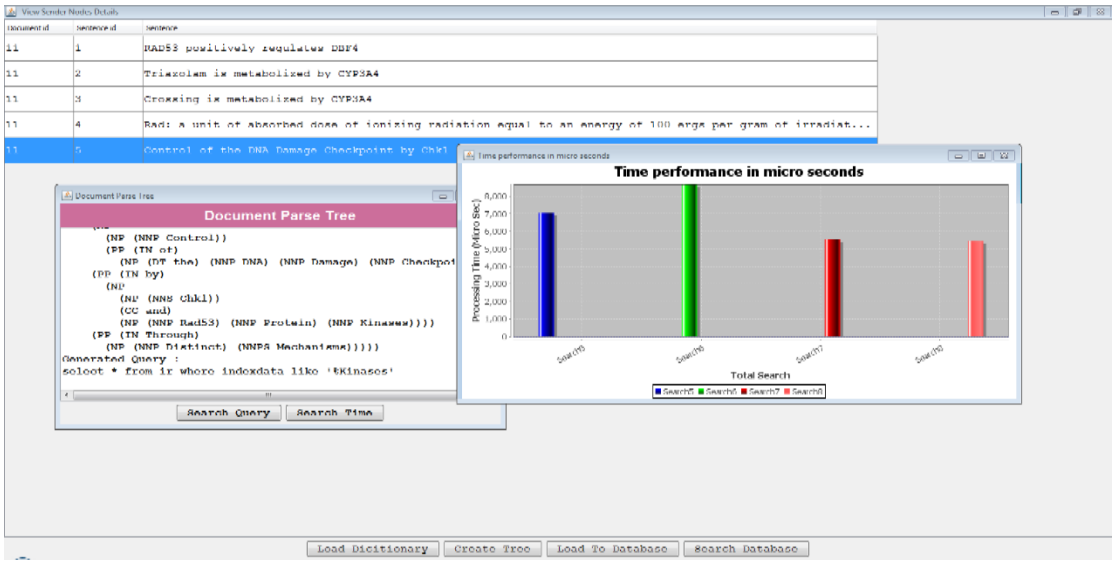


Fig 8.18: Snapshot of Time performance With PTQL

VIII. CONCLUSIONS

Information extraction has been an active research area over the years. The main focus has been on improving the accuracy of the extraction systems, and IE has been seen as a one-time execution process. Such paradigm is inadequate for real-world applications when IE is seen as long running processes. We describe how our proposed extraction framework differs from traditional IE systems, rule-based IE systems and IE systems based on machine learning. While new documents can be added to our text collection, the content of the existing documents are assumed not to be changed, which is the case for Medline abstracts. Our focus is on managing the processed data so that in the event of the deployment of an improved component or a new extraction goal, the affected subset of the text corpus can be easily identified.

The filtering process utilizes the efficiency of IR engines so that a complete scan of the parse tree database is not needed without sacrificing any sentences that should have been used for extraction. Furthermore, our approach

provides automated query generation components so that casual users do not have to learn the query language in order to perform extraction. To demonstrate the feasibility of our incremental extraction approach, we performed experiments to highlight two important aspects of an information extraction system: efficiency and quality of extraction results.

REFERENCES

[1] Ferrucci, David, and Adam Lally. "UIMA: an architectural approach to unstructured information processing in the corporate research environment." Natural Language Engineering 10, no. 3-4 (2004): 327-348.

[2] Tari, Luis, Phan Huy Tu, Jörg Hakenberg, Yi Chen, Tran Cao Son, Graciela Gonzalez, and Chitta Baral. "Incremental information extraction using relational databases." Knowledge and Data Engineering, IEEE Transactions on 24, no. 1 (2012): 86-99.

[3] Chu, Eric, Akanksha Baid, Ting Chen, AnHai Doan, and Jeffrey Naughton. "A relational approach to incrementally extracting and querying structure in unstructured data." In Proceedings of the 33rd international conference on Very large data bases, pp. 1045-1056. VLDB Endowment, 2007.

[4] Cowie, Jim, and Wendy Lehnert. "Information extraction." Communications of the ACM 39, no. 1 (1996): 80-91.

[5] F. Chen, B. Gao, A. Doan, J. Yang, and R. Ramakrishnan,"Optimizing Complex Extraction Programs over Evolving Text Data," Proc 35th ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '09), pp. 321-334, 2009.

[6] S. Bird et al., "Designing and Evaluating an XPath Dialect for Linguistic Queries," Proc 22nd Int'l Conf. Data Eng. (ICDE '06), 2006.

[7] S. Sarawagi, "Information Extraction," Foundations and Trends in Databases, vol. 1, no. 3, pp. 261-377, 2008.

[8] D.D. Sleator and D. Temperley, "Parsing English with a Link Grammar," Proc Third Int'l Workshop Parsing Technologies, 1993.

[9] R. Leaman and G. Gonzalez, "BANNER: An Executable Survey of Advances in Biomedical Named Entity Recognition," Proc. Pacific Symp. Biocomputing, pp. 652-663, 2008.

[10] A.R. Aronson, "Effective Mapping of Biomedical Text to the UMLS Metathesaurus: The MetaMap Program," Proc. AMIA Symp., p. 17, 2001.

[11] M.J. Cafarella and O. Etzioni, "A Search Engine for Natural Language Applications," Proc. 14th Int'l Conf. World Wide Web (WWW '05), 2005.

[12] T. Cheng and K. Chang, "Entity Search Engine: Towards Agile Best-Effort Information Integration over the Web," Proc. Conf. Innovative Data Systems Research (CIDR), 2007.

[13] H. Bast and I. Weber, "The CompleteSearch Engine: Interactive, Efficient, and Towards IR& DB Integration," Proc Conf. Innovative Data Systems Research (CIDR), 2007.

[14] S. Bird, Y. Chen, S.B. Davidson, H. Lee, and Y. Zheng, "Extending XPath to Support Linguistic Queries," Proc. Workshop Programming Language Technologies for XML (PLAN-X), 2005.