IPHV7I3015X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .III
ISSN Online: 2319 – 9253
Print: 2319 – 9245

EXTRACTING EFFICIENT INFORMATION USING RELATIONAL DATABASE QUERIES

[#1]Jagadeeswara Prasad P V -M.Tech Pursuing,
[#2]Ch.Raja kishore -Associate Professor,
[#3]S.Siva Skandha- Assistant Professor
Department of Computer Science & Engineering,
CMR COLLEGE OF ENGINEERING AND TECHNOLOGY, Hyd, TS, India.

Abstract: - Information extraction is the task of finding structured information from unstructured or semi-structured text. It is an important task in text mining and has been extensively studied in various research communities including natural language processing, information retrieval and web mining. It has a wide range of applications in domains such as biomedical literature mining and business intelligence. Two fundamental tasks of information extraction are named entity recognition and relation extraction. The former refers to finding names of entities such as people, organizations and locations. The latter refers to finding the semantic relations such as Founder Of and Headquartered In between entities. In this chapter we provide a survey of the major work on named entity recognition and relation extraction in the past few decades, with a focus on work from the natural language processing community.

Keywords- Text mining; query language; information Storage and retrieval.

I. INTRODUCTION

The purpose of information extraction (IE) is to find desired pieces of information in natural language texts and store them in a form that is suitable for automatic querying and processing. IE requires a predefined output representation (target structure) and only searches for facts that fit this representation. The area of information extraction needs to implement methods for attaching structured data from language text which was natural. Normal of structured information is the extraction of entities and relationships of data in between entities. Information extraction is critically shown as a one-time process for the extraction of a specific kind of relationships of well-known from a collection of document. This would also require extraction to be performed from scratch on the entire corpus. However, we observe that only a portion of the corpus is affected with newly recognized entities, as the majority of the entities are overlaps between the original and the improved recognizers. Such expensive re-computation should be minimized. This is particularly true for extraction in the biomedical domain, where a full processing of all 17 million Medline abstracts took about more than 36K hours of CPU time using a single-core CPU with 2-GHz and 2 GB of RAM. In this case, the Link Grammar parser [3] contributes to a large portion of the time spent in text processing.

In this paper, we propose a new paradigm of information extraction in the form of database querie. We present a general-purpose information extraction named as pseudo relevance feedback driven query generation in the context of biomedical extraction. Which can efficiently handle diverse extraction needs and keep the extracted information up- to-date incrementally when new knowledge becomes available? Our proposed information extraction is of two phases.• Initial Phase: In initial phase we perform a one-time parse, entity recognition and tagging is applied on the whole corpus based on current knowledge. The generated syntactic parse trees and semantic entity tagging of the processed text is stored in a parse tree database (PTDB).

• Extraction Phase: Extracting particular kinds of relations can be done by issuing an appropriate query to PTDB. As query languages such as Path[3] and XQuery[4] are not suitable for extracting linguistic patterns [2], we design and implement a query language called PTQL for pattern extraction which effectively achieves diverse IE goals [6].Queries are issued to identify the sentences with newly recognized mentions. Then extraction can be performed only on such affected sentences rather than the entire corpus. Then, we achieve incremental extraction, which avoids the need to reprocess the entire collection of text unlike the file-based pipeline approaches.

There are several advantages of the proposed approach, which have been demonstrated in our initial experimental evaluation. First, using database queries instead of writing individual special-purpose programs, information extraction becomes generic for diverse applications and becomes easier for the user. The user can express and analyze an extraction pattern by issuing a database query. When a user has a new extraction goal, the user only needs to write another query on PTDB without developing and running new programs.

Second, upon new extraction goals, the two-phase approach avoids performing the initial phase again, an extremely expensive phase that has be to performed by existing approaches. We propose an algorithm which automatically generates PTQL queries from training data. To store intermediate text processing output in a database we built parse tree database whose goal is to minimize the need of reprocessing the entire collection of text with the help of old and newly generated data.

We highlight the contribution of the following paper:

An Algebraic Approach to Rule-Based Information Extraction:

In this paper an algebraic approach is used for rule-based information extraction which addresses the scalability issues through query optimization. Rule-based programs are developed which are more powerful then diverse data sets. the main goal is to extract text specific characteristics of any operator.

Prioritization of Domain Specific web Information Extraction:

It is often desirable to extract structural information from row web papers for better information browsing , query answering and pattern mining. this this paper the prioritization approach is used in which any candidate pages from the corpus are ordered according to their expected contribution to the extraction results in higher estimation of its value.

GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications:

This GATE is very important survey to our paper as, it basically provide graphical development environment for robust natural language processing. It also process human language. An infrastructure is developed and deployed which will in return help the user in three ways:

By specifying an architecture, or organizational structure for language processing software. It also provides a development environment built on the top of the framework which is made up of convenient graphical tools for developing components.

II. RELATED WORK

Figure 1 illustrates the system architecture of our PTQL framework. The Text Processor performs the Initial Phase for corpus processing and stores the processed information in the Parse Tree Database (PTDB).

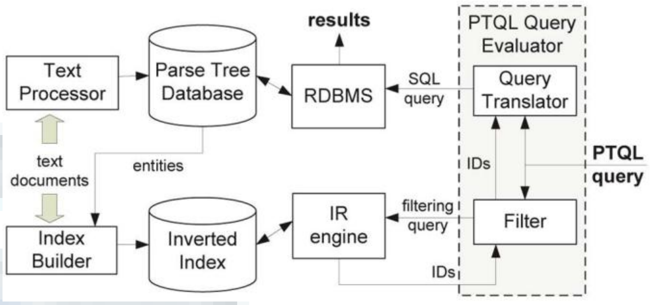

Fig. 1. Architecture of PTQL framework. The framework includes the parse tree database for storing intermediate processed information and the query evaluator for the evaluation of PTQL queries through filtering and translation to SQL queries

After the text processor the parse tree database(PTDB) is responsible for the entire corpus processing and storing the processed information into it. Then comes the extraction phase , where PTQL query evaluator takes a PTQL query and transforms it into keyword-based queries. to speed up query evaluation, index builder creates an inverted index for indexing the sentences according to words. The incremental information extraction using relational database queries is based on the pseudo- relevance feedback driven approach that takes keyword- based queries, and the PTQL query generator then finds common grammatical patterns among the top-k retrieved sentences to generate PTQL queries.

Link Grammar

The Link Grammar parser is a dependency parser based on the Link Grammar theory [7]. Link Grammar consists of a set of words and linking requirements between words. For a sentence it has given some functionality which it should satisfy:

1) A sentence is sequence of words. 2) Sentence links should not cross.

3) The words form a connected graph, and

4) The links satisfy the linking requirements of each word in the sentence.

Link grammar is capable of producing constituent trees. A constituent tree is a sentence with the node represented by parts of speech tags and words of sentences in the leaf nodes.

A Leaf node of constituent tree represents the words of the sentence and their parts of speech tags.

Let's consider an example for the sentence "RAD53, which activates DNA damage, positively regulates the DBF4 protein".

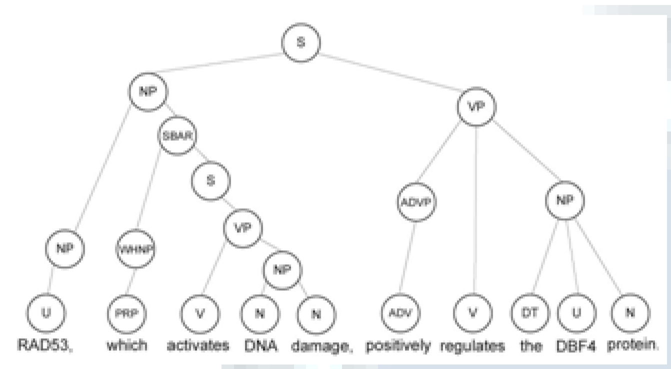

Fig. 2. Constituent tree of the sentence "RAD53, which activates DNA damage, positively regulates the DBF4 protein."

Parse Tree Database (PTDB)

A parse tree is an ordered rooted tree that represents the syntactic structure of a string according to some formal grammar.

The Text Processor parses Medline abstracts with the Link Grammar parser [3], and identifies entities in the sentences. Each document is represented as a hierarchical representation called the parse tree of a document. A parse tree is composed of a constituent tree and a linkage.

Figure 2 shows a sample parse tree of a sentence, where the solid lines indicate parent-child relationships in the constituent tree and the dotted lines represent the linkage. A linkage represents the syntactic dependencies between pairs of words in a particular sentence.

Inverted Index

Inverted index is an index data structure storing a mapping from content, such as word or numbers to its location in a database file or in a format of documentation or a set of documents. The main propose of inverted index is to allow fast full text search at a cost of incremented processing when a document is added to the data base. This inverted index enables the efficient process- sing of PTQL queries. Inverted index takes a number of documents and we apply stop words on it and then result is produced in tabular format which will Midterm and documents. we parse a document through each document and extract tokens. these tokens are stored in an hash map(as a key) and the total number of times they occur in the collection (i.e. frequency) is stored as the value.

Parse tree Query language

To perform information extraction, we propose a query language, PTQL, to specify linguistic patterns on parse trees. A PTQL query consists four components delimited by colons: (i) tree patterns, (ii) a link condition, (iii) a proximity condition, and (iv) a return expression. A tree pattern describes the step-by-step structure and the horizontal order of the nodes in a linguistic extraction pattern. a link condition also occur which represents the syntactic dependencies or any kind of links between pair of words in a sentence.

A PTQL is made up of four components they are

1) tree patterns,2)link conditions,3)proximity conditions, and 4)return expression

A proximity condition specifies words that are within a specified word distance in the sentence. A return expression defines the list of elements to be returned.

III. IMPLEMENATION

Initial Phase

Sentence Splitting: In the first module the documents contain sentences. The sentences are in the unstructured manner. The module converts sentences to structured sentences with index. This process is applied on the existing corpus.

Word Indexing: Each sentence of a document is made up with different words.

Example: S1={w1,w2,w3…….wn}

Document	Sentence	Word index	Word
Doc1	1	1	Hello
Doc1	1	2	world

The module splits all the indexed sentences by words.

Word Tagging: The words will be presented in the document in different forms such as present, past, future. The words has to be n-grimed to find out the possible equivalence of root words. The root words can be grouped together (or) clustered for special group of interests. Example: {“cricket”, “football”} can be grouped together to special interests called “sports” category. Identifying group of words of similar category can have relationship.

Parse Tree Database (PTDB) Construction: The word-net is a semantic relational network. The word-net is store in the database as PTDB.

User’s Query Preprocessing: User’s query has to be preprocessed against stop words elimination. The query words has to be n-grammed for possible root words.

Query Word Tagging (PTQL):All the n-grammed words may not be the root words All the n-grammed words may not be the root words. Find the semantically words for each word of query root word. Find the appropriate Tag with their relevancies (or) Frequencies.

Query Evaluation and Query Generation

The parse tree query language query evaluator will take a PTQL and convert it into key-word based query and also in the form of SQL queries, which are evaluated by the underlying the RDBMS and information retrieval.

There are two types of query generation which will automatically generate PTQL: training set driven query generation and pseudo-relevance feedbackdriven query generation. In this paper Pseudo-relevance feedback driven query generation. The idea behind it is to automatically generate PTQL[9] by considering the constituent tree with Boolean key based query. The Boolean keyword based will be consisting of number of query term, where a query term can be any form of keyword, or an identifier which will provide the semantic name of a protein names. It always starts its processing from top-to-bottom whereas Training set driven Query generation will do its generation process manually and in the same manner annotates data to generate PTQL for the extraction of protein-protein interaction. A this methodology have many disadvantage as the training data is not always available.

PARSE TREE GENERATOR

This Module provides to create the parse tree database for bio-medical information. The parse tree is used to extract the large amount of texts and multiple sentences (per-sentence) at a time. Parse tree database is storing the documents in the form of parse tree and also provide the ability to compute statics across multiple extraction for booting the confidence of extracted fact.

IV. IE APPROACHES

The main focus so far has been on improving the accuracy and runtime of information extractors. But recent work has also started to consider how to manage such extractors in large- scale IE-centric applications. Our work fits into this emerging direction, which is described. While we have focused on IE over unstructured text, our work is related to wrapper construction, the problem of inferring a set of rules (encoded as a wrapper) to extract information from template-based Web pages. Since wrappers can be viewed as extractors, our techniques can potentially also apply to wrapper contexts. In this context, the knowledge of page templates may help us develop even more efficient IE algorithms

Traditional IE Approaches:

Popular IE frameworks such as UIMA[1] and GATE [2] provide the ability of efficient integration of various NLP[8] components for IE. Such frameworks are file based and The intermediate processing data are the inserted into the parse tree database so that both the new and existing processing data can be utilized for extraction.

While considering time factors, the incremental approach shows the more data extraction compared with Existing approaches.

Incremental information extraction approach

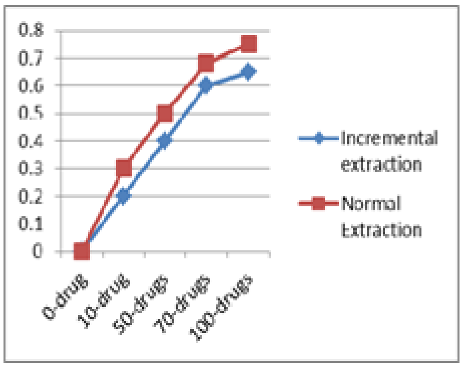

Fig 3.1 Time Performance for extraction

As figure 3 shows the improvement in the efficiency in the time process.as by using our traditional approach it takes around 36 K hours but this approach will reduce most of the time by 89.64% so in result it will be more efficient and process any large data sets.

V. CONCLUSIONS

The existing extraction frameworks do not provide the capabilities of managing intermediate processed data such as parse trees and semantic information .This leads to the need of reprocessing of the entire text collection, which can be computationally expensive. With the use of parse trees, our extraction framework is most suitable for performing extraction on text corpus written in natural sentences such as the biomedical literature. The increment extraction approach saves much more time compared to performing extraction by first processing each sentence one- at-a time with linguistic parsers and then other components. This comes at the cost of overheads such as the storage of the parse trees and the semantic information, which takes up 1.5 TB of space for 17 million abstracts for the parse tree database. In the case when the parser fails to generate parse tree for a sentence, this system generates a "replacement parse tree" that has the node STN as the root as the words in the sentence as the children of the root node. so for this for future enhancement we can use the map reduce techniques so that it can help the user to process large volume of data and increase the efficiency and it will also reduce the time and increase the performance and it will also reduce the search space between the words which the traditional approach will not provide it with map reduce the overall performance of the system will be increase and it will be user friendly too.

REFERNCES

1)D.Ferrucci and A.Lally, "UIMA:An Architectural Approach to Unstructured Information Processing in the corporate Research Environment",Natural Language Eng.,vol. 10,nos. ¾,pp.327-348,2004.

2)H.cunningham, D. Maynard, K. Bontcheva, and V.Tablan, "GATE : A Framework and Graphical Devlopment Environments for Robust NLP Tools and Applications",Proc.40th Ann. Meeting of the ACL, 2002.

3)J.Clark and S.DeRose, "XML Path Language(XPath)," http:// www.w3.org/TR/xpath, Nov. 1999.

4)"XQuery1.0:An XML Query Language," http://www.w3.org/ XML/Query, June 2001

5) D. Grin berg, J. Lafferty, and D.Sleator, "A Robust Parsing Algorithm for Link Grammars," Technical Report CMU-CS-TR-95-125, Carnegie Mellon Univ. 1995.

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

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

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

9)E.Agichtein and L.Gravano,"Querying Text Databases for Efficient Information Extraction,"Proc.Int'l Confidante (ICDE), pp. 113-124, 2003.