Sindh Univ. Res. Jour. (Sci. Ser.) Vol. 48 (2) 233-240 (2016)

SINDH UNIVERSITY RESEARCH JOURNAL (SCIENCE SERIES)

Using Machine Learning to Generate SPARQL Queries from NL for NoSQL Database

I. S. BAJWA++, F. RAZZAQ*, A. H. S. BUKHARI**, R. AMIN***

Department of Computer Science, The Islamia University of Bahawalpur

Received 15th December 2015 and Revised 2nd May 2016

Abstract: SPARQL (SPARQL Protocol and RDF Query Language) is a RDF (Resource Description Framework) query language for "key-value" databases such as NoSQL databases e.g. MongoDB. In the recent times, SPARQL/RDF has emerged into a powerful way of extracting data from database columns that contain multiple values for the same key, and especially wherever the columns are joinable variables in the query. However writing a SPARQL/RDF query is in itself a challenging task since SPARQL/RDF' syntax is of descriptive and declarative nature and requires technical skills.A problem in applying SPARQL/RDF queries to NoSQL databases is the extraction of irrelevant results without sensing context of given piece of input against a single query. In such cases, a user cannot get the required information from linked data in the same context as wanted. To address such problem, there is need of an intelligent approach that sense the context of given piece of natural language input and generate context oriented output to end user in the form of a SPARQL/RDF query. In this paper, a natural language query based framework is presented to generate SPARQL/RDF queries for NoSQL databases. Here, a challenging task is to map natural language queries to map to SPARQL queries. Our framework accepts input natural language query in English from user and then translates the NL queries into SPARQL/RDF queries. The results of the experiments with the deigned framework reflect importance of such approach used for automated generation of SPARQL/RDF queries for NoSQL databases.

Keywords: NoSQL Databases, SPARQL, Machine Learning, Natural language queries

1. INTRODUCTION

In the last decade, RDF database systems have emerged into standardized NoSQL solutions available at the moment (Collobert, 2011). Being built on a simple, uniform data model and a powerful, and a declarative query language. In modern information system, RDF based Linked Data (Alexander, *et al.,* 2013) is growing at surprising rate. One of the current challenges in accessing and consuming Linked Data on the web is dealing with difficult user interface. Since, data can be distributed and duplicated in different data stores, and described by different vocabularies and as a result a user may face difficulties in writing query for linked data (Sujatha, *et al.,* 2012; Kaur, 2013). As a result, accessing and consuming Linked Data on the Web requires end users to be aware of where to find the data (which datasets) and how it is represented (which vocabularies). Hence, users need to have working knowledge of formal query languages such as SPARQL to access the datasets. This scenario creates high barriers of entry in creating applications that uses Linked Data (Bouhali, 2015). For users to express their information in simple and convenient way they need the usage of Natural Languages Queries (NLQs). Therefore, we assume that enabling a user to use Natural Languages to query Linked Data would offer a favourable alternative to formal query languages. The first step in translating to a SPARQL is to map the keywords in the query to Linked Data resources. Natural Languages queries emerged as a straightforward and perceptive way for users to query Linked Data (Bizer, *et al.,*2005; Djahantighi, *et al.,*2008)

It is always a complex and professional problem in using query language for database. There is a limit in usage of existing data from data base because it causes the complexity but in past there exists a lot of software that uses for in executing lexical parse semantic analysis on natural language sentences to transform it into SQL (Structured Query Language). Just like natural language interface database (Orsi, *et al.,* 2011; Naeem, *et al.,* 2012, Sordoni, *et al.,* 2015). Other already implemented systems have been surveyed and result of all was same as to natural language processing systems. Here we will research for Natural language based on context aware query generation for linked data that find results according to analysing the context of query (Jurafsky, *et al.,* 2000). Data publishing and integration on the Linked Data web differs from traditional integration systems as there are no central components or coordinators. The integration tasks are split up and spread across multiple participants: separate systems and people are responsible for data generation, publishing, mapping/linking and querying.

++Corresponding authors email: imran.sarwar@iub.edu.pk, ahsbukhari02@gmail.com
*Department of Computer Science, NCBA&E, Lahore
**Sindh Institute of Management and Technology, Gulistan-e-Johar, Karachi
***Baluchistan University of IT, engineering and Management Sciences, Quetta

2. RELATED WORK

Natural Languages processing got a certain attention in last few years. An approach was developed to interact with database by Alexander (Alexander, *et al.*,2013)it was a great effort to minimize the gap in communication between computer and human. To retrieve information from database it requires knowledge about query language such as SQL but most of the user has not technical knowledge about SQL. After expert system NLIDB is a Natural Languages Web Interface for Database (NLWIDB) has been developed early database systems was LUNAR, LIFER/LADDER and English wizard are the examples of NLP database systems. NLWDB facilitates the user to communicate with database over the internet. Natural Languages Web Interface Database is domain independent in accessing the information. Let's takes an example of university database that contains many tables in an organized way but when a person wants to access data from globally he will take the help from SQL language to access the database but what will happened when a person not to use a technical language he just writes the question in Natural Languages in browser the result will be the same as SQL language and Natural Languages. In NLWIDB has following modules in its system GUI, word check, tokenization excess words remover, mapping rules and SQL element identifier so NLWIDB is concerns with translating user query from English to SQL query language to retrieve data from database. Over the internet, an algorithm was developed that maps the Natural Languages question to SQL statement this algorithm has been implemented with Apache, PHP, MYSQL (Collobert, 2011). Natural Languages understanding are much harder than Natural Languages generation so far NLIDB is an expert system that first parser the given input then change it into Natural Languages expression to SQL language. A number of ways to query data in natural languages are discussed in (Sujatha, *et al.,* 2012). Information playing a vital role now days in our daily life and a major source of information is a database. This system is implementation of intelligent natural languages interface for a database providing a simple way of querying to a user. Similarly, natural language processing is also used in querying data from web (Kaur, 2013). De to the rapid proliferation of Linked Data on the Web, the topic of Linked Data query processing has recently gained attention. Works in this domain does not assume full availability of SPARQL endpoints. Since, data dictionary and data manipulation languages are mainly used to handle with ambiguity in information extraction. Djahantighi (2008) discussed sources of complexity in querying data such as ambiguity and multiple senses of a query. An interface for natural languages composing statistical parsing with semantic tractability was proposed. The PERICES interface plug in is a statistical parser. This work enables the PRICES interface to overcome the parser error and map correct parsed question to SQL (Bajwa, 2008). To overcome these errors in parsing strong semantics model paired with light re-training. Statistical parser database schema independent and give 94% accuracy with Benchmark ATIS datasets. Another interface was purposed for query in XML database. Natural Languages interface for querying on XML database. It takes random English language sentences as input query translates input query into XQuery expression for mapping XML database is done via grammatically proximity of Natural Languages parse token to exact values of elements in the result. However, no support is available to generate SPARQL queries from natural language. (Bunningen 2008) presented an automated system for providing a natural language interface to ontologies. This system is a good example of natural language supported systems. However, this system didn't support NoSQL queries. Similar tools are Auto-SPARQL (Lehmann, and Bühmann, 2011) and NL-SPARQL (Sharef, and Noah, (2013), however they lack of NoSQL Database support. The discussed work shows a major gap in research and we address this problem in this research paper.

3. USED APPROACH

Natural Languages processing is the primary mean to communicate anywhere in the world. Natural Languages are much ambiguous and have a lot of semantics against a single word so far to retrieve useful data (Jurafsky, *et al.,* 2000) it processed before to get actual data from resources because Natural Languages are syntactical ambiguous and semantically inconsistence and unaware from context of NL query. To solve this problem a framework is developed to generate a Context Aware query from NL query to retrieve the concert result from linked data. The proposed system **(Fig. 3.1)** can generate Context Aware query for linked data using Context Analysis. This approach will take processed Natural Languages to generate Context Aware query. This interface take input (English Language text) and transform Natural Languages query into Context Aware query with the help of Context Analysis. The generation of Context Aware query from Natural Languages for linked data it involves three basic steps.

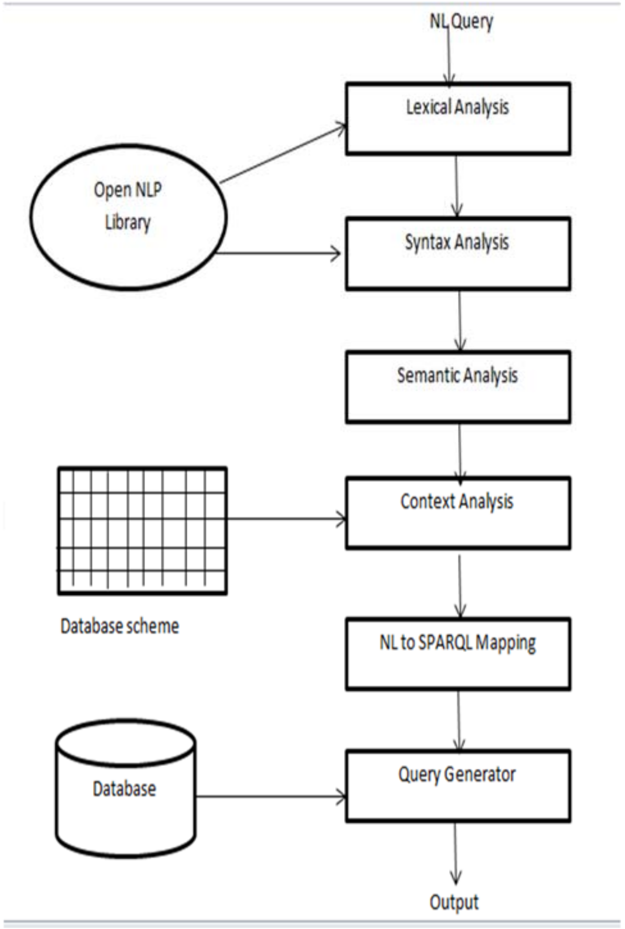


Fig.3.1: Stages in Context Analysis query for linked data through NL

3.1Lexical Analysis

The NL parsing starting phase is lexical analysis simple (input) text consists on English language wording. Lexical analysis is done by sentence splitting morphological analysis tagging parts of speech (POS). Valid tokenization, lexicons or symbols are generated in a lexical processing (Jurafsky, *et al.,* 2000).. These tokens are used for further processing text. When input is give system breakdown into tokens for examples "Fareeha goes to school". It will tokenize as [Fareeha] [goes] [to] [school] [.]. After tokenization phase the next phase is splitting sentence in which margins of sentence recognized and placed separately in array list. Next phase is Parts of speech tagging in which previously generated tokens are labeled as verb phrase nouns pronoun adjective helping verb preposition conjunction interjection etc. there are many other categories of noun and pronoun. Stanford parser v 3.0 can identify 44 POS tags (Jurafsky, *et al.,* 2000).. Morphological analysis analyze the individual words it is the description of structure of given language morphemes and other basic units of language such as root words prefix suffix affixes and circumffix.

3.2 Syntactic Analysis

Syntactic analysis focused on the structure of the given input text. The structure of sentence consist phrase hierarchy. Syntactic analysis transforms words into a form in which it shows that how the words are associated with each other in a sentence. Determining the word association a parse tree can be generated for further processing. In our approach it will be done with predefined library "Open NLP" that process Natural Languages doing tokenization of English language strings lama's morphemes hyponymy, meronymy, etc. are all checked here tagging and word grouping also done here.

For example

Customer deposit RS 1000 in Bank account.

Word grouping can be break here just like

[Customer] [Deposit] [Rs 1000] [In] [bank_account].

Bank account is a group of words in this example; we use it with the help of '_' special symbol.

3.3 Semantic Analysis

Semantic analysis module is related to acquisition of meanings. Meaning of the sentence is composed on words used in the sentence. For this purpose predefined library semantic measure is used to analysis of the sentence parts that has been broken by parser in pre phase syntactic analysis. Semantics are check to define the meanings; one word has many meanings of same word so lexical semantics are check here. Just like a word "Bank" has two meanings either bank organization or bank of a river. Therefore sense of meaning may miss lead to irrelevant result of query so it is necessary to do lexical analysis to driven a Context Aware query for linked data in database. Semantic role labelling is done in this phase for example: Customer deposit RS 1000 in Bank account. Customer is table name, Deposit is field name, Rs. 100 is value and bank account is also table name.

3.4 Context Analysis

Context basically determined the sense or give meaning of particular situation. To get answer through Natural Languages query it is quite ambiguous and does not generates concrete results. Linked data is published on web in very huge amount. So to avoid unnecessary results we are going to adopt an approach that focused on actual results by translating Natural Languages query into Context Aware query so that unwanted data may avoid and save time in acquisition irrelevant linked data on the web. For this method to query Context Aware data, database schema is used to formulate appropriate query to get the desire result from database. We can improve query efficiency and precision by focusing search only meaningful data according to the current context. This approach is purely based on semantic technology and supported by Context Awareness.

Semi-supervised classification algorithms have been performing better for natural language text classification to NoSQL elements. Semi-supervised algorithms such as the Expectation-Maximization (EM) (Sujatha,*et al.*,2012)successfully used for classification of NL text documents.EM is one of the iterative algorithms that are capable of completing missing data. EM typically computes the expected value for each missing value in data based on maximum likelihood estimation of existing values in the data and such expected values fill the missing values in the data. Such ability of EM algorithms makes it suitable for classification of NL data that is inherently semantically incomplete.

Input of EM approach is a collections R_l of labeled SBVR rules and R_u of unlabeled SBVR rules. The EM algorithm is applied in two stages: Expectation stage, and Maximization stage. In Expectation stage, the missing data is filled in and in the Maximization stage, the parameters are estimated on the basis of filled data. To perform these two stages, the training of naive Bayesian classifier performed as shown in equation (1). For training, the labeled examples used only that becomes initial phase of EM and then this process is iteratively repeated.

$$P(c_j \mid v_i) = \frac{P(c_j)\prod_{k=1}^{|v_i|} P(v_i \mid c_j)}{\sum_{r=1}^{|C|} P(c_r)\prod_{k=1}^{|v_i|} P(v_i \mid c_j)} \qquad (1)$$

Here, *R*is a set of SBVR rules that are representation of software requirements and an ordered list of vocabulary items. Each vocabulary item in a SBVR rule r_i is represented byv_i, where SBVR rule is list of vocabulary as $V = <v_1, v_2, \dots, v_{|n|}>$. Here, the vocabulary items are classified into target UML class model elements. In this paper, the possible types of UML class model items are denoted as classes, $C = \{c_1, c_2, \dots, c_{|n|}\}$. Here, classification s performed by computing the posterior probability, $P(c_j \mid R_i)$, where *cj*is a class and *vi* is a SBVR vocabulary item.

For the maximization phase, equations (2) and (3) are used, where equation (2) represents the Bayesian probability and the multinomial model.

$$(2) \qquad P(c_j) = \frac{\sum_{i=1}^{|R|} P(c_j \mid v_i)}{|R|}$$

In equation (3), the Laplacian smoothing is used, where $N(v_t, r_i)$ is the total number of times the vocabulary item $v_t$appears in a rule $r_i$and where $P(c_j \mid r_i)$□{0,1, 2, **… .** *n*} depends on the class label of a vocabulary item. Here, the naive Bayesian classifier helps in identifying the class with the top $P(cj|di)$ value and that class is assigned to that vocabulary item.

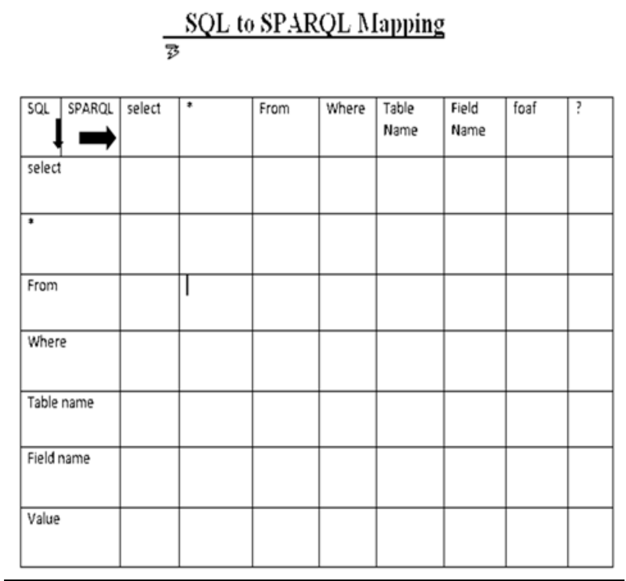
$$P(v_i \mid c_j) = \frac{1+\sum_{i=1}^{|R|} N(v_t, r_i)\, P(c_j \mid r_i)}{|V| + \sum_{z=1}^{|V|}\sum_{i=1}^{|R|} N(v_z, r_i)\, P(c_j \mid r_i)} \qquad (3)$$

Here, an important issue is raised in [3] where it is investigated that the basic version of EM works for only where there exists one–to–one relationship in classes and mixture components. Since, in NL, there can be a few cases where a class can be associated with multiple mixture components and such situation leads to a one-to-many relationship in classes and mixture components. It is discussed in (Kaur, 2012) that multiple mixture components M-EM performs better than basic EM with text classification. In our approach, we experimented with both variations of EM approach.
Here in this phase keywords are matched in database schema for context checking how the query is working there are many keywords just like select insert delete update, etc.

3.5 SQL to SPARQL Mapping

SQL query is used from relation database while SPARQL is used for RDF graph data on the linked data. Here are some keywords that are used in both query languages some are same in both and some are different keywords. The mapping from SQL to SPARQL is shown in **(Fig. 3.2).**

SQL Query:	Select * from person where name= “NAME” and email=”mail”
SPARQL Query:	Select * where { ? person foaf: ? name. ? person foaf: email ? mail }



SQL to SPARQL Mapping

SQL ↓ SPARQL ➡	select	*	From	Where	Table Name	Field Name	foaf	?
select								
*								
From								
Where								
Table name								
Field name								
Value								

Fig. 3.2:-Mapping from NL to SPARQL

3.6 Query Generation

After context analysis, the chunks that were derived from database schema is further goes to query generator module where SPARQL query is generates for linked data and mapped into database and shows the output result to end user.

The proposed approach for context analysis of the Link Data queries written in SPARQL is discussed in the previous thesis. In this thesis, details of the experiment settings and results of the preliminary experiments are discussed in this thesis. The following section contains a case study that has a set of English queries and these English queries are contextually analysed using the proposed approach.

4. EXPERIMENTS AND RESULTS

The case study done in this thesis is based on car rental database schema **(Fig. 4.1).** The database schema contains five tables and a set of English queries. Database Schema basically is a structure that describes the internal representation of database. It is support by database management system. It is divided into database tables in case of relational database. The formal definition of database schema is set of formulas. The database describes that how real world entities are mapped in database. The schema defines the tables, fields, relationship, views, indexes, function and trigger etc. Schema stored in data dictionary and term sometimes used graphical representation of database structure. The used database schema is shown in Figure 4.1 that has four tables; Cars, *RentalOrders*, *Employees*, *Customers*, *RentalCars*. This diagram describes a schema of rental cars. It consists of five tables; Customers, RentalOrders, Employees, Cars and Rental Rates. Let's execute a query with an example: Customer want a car on rent, how the search process will be done and find out the best possible outcome after context analysis.

Step1. Natural Language Query: Customer/ User will query from the interface in Natural Language that may be like this, the list of companies' car rented.

Step2. Lexical Analysis: In Lexical analysis process chunking / Tokenization id done. The list of companies' car rented.

[The] [list] [of] [companies'] [car] [rented] [.]

The	**→ Determiner**
List of Companies	**→ NNPS**
Car	**→ NN**
Rented	**→ VBN**

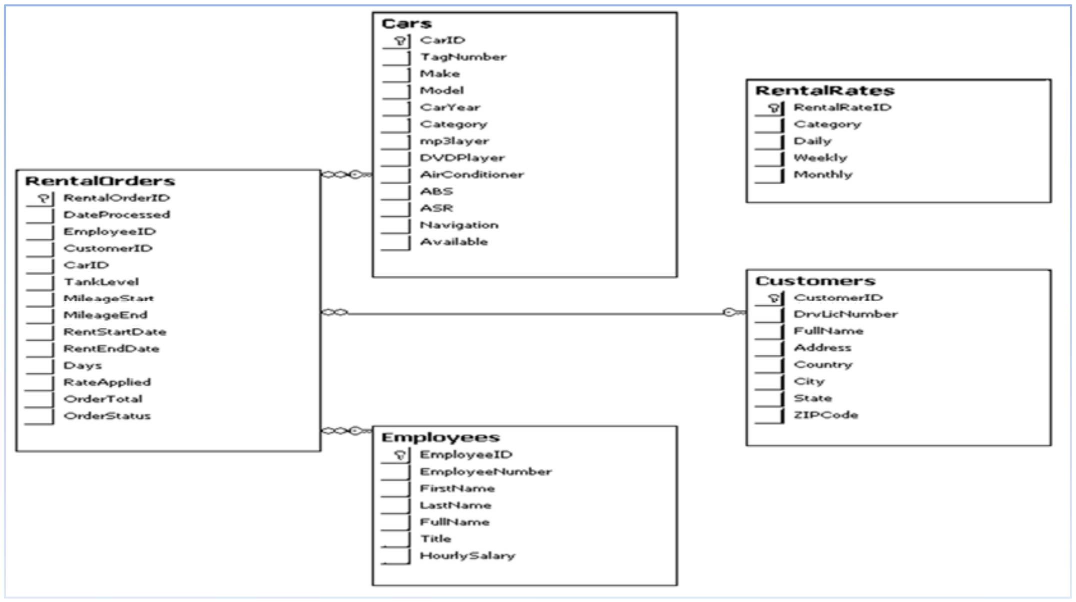
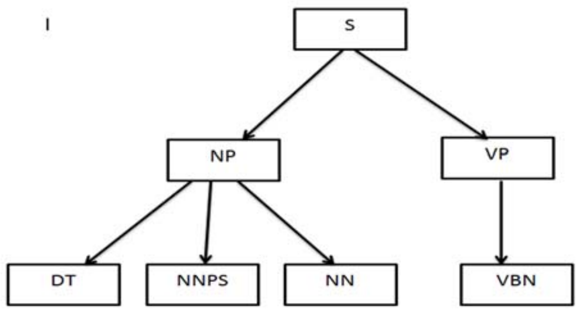


Fig. 4.1: Database Schema

Step3. Syntax Analysis: In syntax analysis process, identification of sentence structures will be done so that syntax tree draw. Word grouping also is done on this stage. An example of the detailed syntax analysis is shown in **(Fig. 4.2).**

Example [The] [List of companies'] [Car] [Rented] [.]
Parse tree:

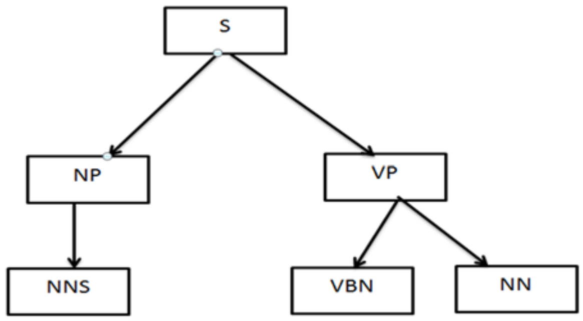


The list of companies' car rented
Fig. 4.2:Syntax Parse Tree

Step4. Semantic Analysis: The semantic analysis process will generates the semantic with the help of semantics role labelling

Syntax-Driven Semantic Analysis
"Cars rent companies"

Step5. Context Analysis: Database schema will match these words that it contains previously record match with this new NL query and pass it to next process. An example of detailed syntax analysis is shown in **(Fig.4.3).** Cars is a name of table in database previously stored are Rental is also table name in database so it pass to next step.



Cars rental companies
Fig. 4.3Semantic Tree

Step6. Natural Language to SPARQL Mapping:
NL → "Cars rent companies"
SQL → Select *car from cars where car id= "1".
SPARQL → Select * where
{
? Cars foaf car id= "1"
}

Step7. Query Generator: In last step the database software generates query according to the SPARQL and display the result against car id to the end user.

Example 2: Following is second running example.

Step1. Natural Language Query: Rent car for 24 hours.

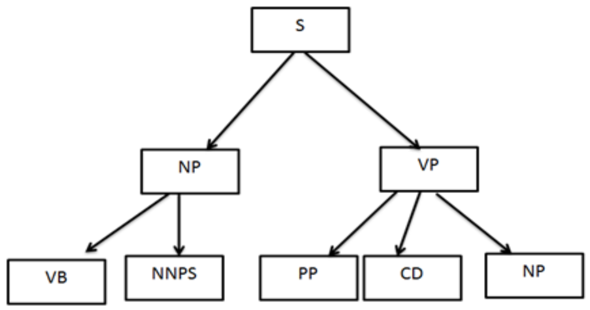
Step2. Lexical Analysis: In Lexical analysis process chunking / Tokenization are done. Rent car for 24 hours

[Rent][Car] [For] [24] [Hours] [.]

Rent	→ VB
Car	→ NN
For	→ PP
24	→ CD
Hours	→ NPS

Step3. Syntax Analysis: In syntax analysis process, identification of sentence structures will be done so that syntax tree draw. Word grouping also is done on this stage.

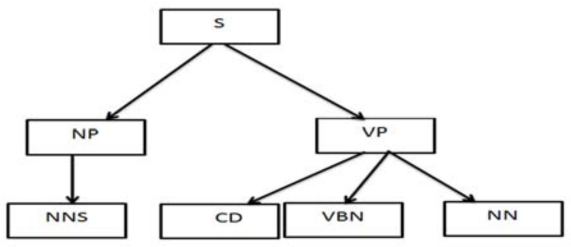
[Rent][Car] [For] [24] [Hours] [.]
Parse tree:



Rent cars for 24 hours

Fig.4.4 E2 Pars Tree

Step4. Semantic Analysis: The semantic analysis process will generates the semantic with the help of semantics role labelling
Syntax-Driven Semantic Analysis
"Cars on rent for 24 hours"



Cars for 24 hours on rent
Fig. 4.5 E2Semantic Tree

Step5. Context Analysis: Database schema will match these words that it contains previously record match with this new NL query and pass it to next process. Cars is a name of table in database previously stored but no other keyword is matching in database schema.

Step6. Natural Language to SPARQL Mapping:

NL →"Cars for 24 hour on rent"

SQL → Select *car from cars where car hours= "24".

SPARQL → Select * where

{

? Cars foaf car hours= "24"

}

Step7. Query Generator: In last step the database software generates query according to the SPARQL and display the result against car hours not found in database schema to the exact result will not show to the end.

Table 4.1: Results of the preliminary Experiments

Examples	**Total queries**	**Correct**	**Incorrect**	**Accuracy**
E1	4	3	1	75%
E2	6	4	2	67%
E3	5	3	2	60%
Total	**15**	**10**	**5**	**61%**

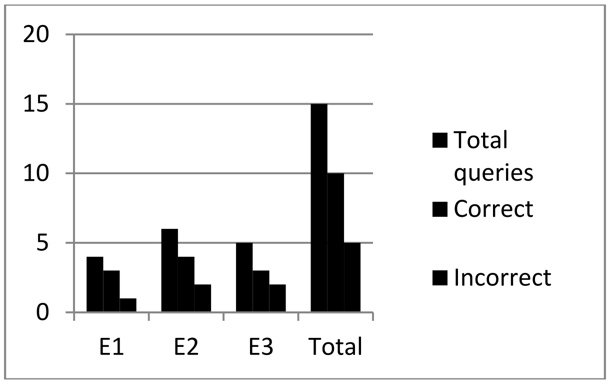


Fig. 4.6: Results of contextual analysis of NL Queries

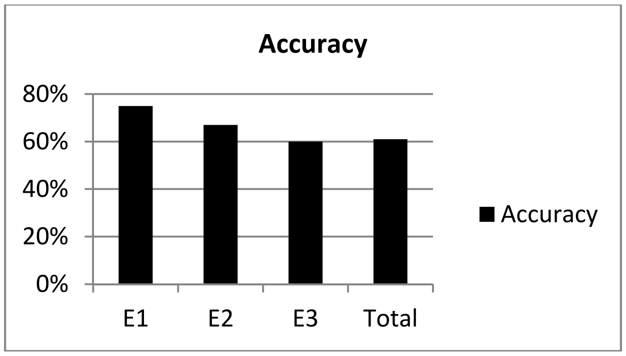


Fig. 4.7: Overall Accuracy of contextual analysis of NL Queries

The case study is done with the entire step used in proposed approach. Result and evaluation is checked with the help of natural language queries examples. The results of the experiments are tabulated in **(Table 4.1)** and depicted in **(Fig. 4.6)** and accuracy of each example is showed in the Figure 4.7.Accuracy of example1 at level 75%, example2 at level 68% and accuracy of example3 is at level 60%. At the end we can say accuracy of this proposed work is approximate at level 62%.

5. CONCLUSION

This thesis describes and elaborates the steps that are used in this methodology. To achieve Context Analysis for Natural Language Query for linked data the process starts from Lexical Analysis, Syntactic Analysis, Semantic Analysis and Context Analysis. With the help of this process we become successful to achieve the target to generate Context Analysis of Natural Languages for linked data. Most of the information given in the sentence structure is clear to understand the semantics of the Natural Languages question and no need of much information for querying on available data but building a complete sentence and transform it into a query it require more time to obtains desire data.

REFERENCES:

Auer, S., C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, (2007). Dbpedia: A nucleus for a web of open data 722-735. Springer Berlin Heidelberg.

Alexander, R., P. Rukshan, S. Mahesan, (2013) Natural Languages Web Interface for Database . 3rd international symposium Oluvil Sri Lanka 7th July, 2013

Bajwa, I. S., S. Mumtaz, and M. S. Naveed, (2008). Database Interfacing using Natural Language Processing. European Journal of Scientific Research, 20(4), 844-851.

Bizer C., T. Heath T. Berners-Lee, T. Linked-Data (2009). The Story So Far. International Journal on Semantic Web and Information Systems (IJSWIS) 5, 3, 1-22

Bouhali, R., and A. Laurent, (2015). Exploiting RDF Open Data Using NoSQL Graph Databases. In Artificial Intelligence Applications and Innovations 177-190. Springer International Publishing.

Collobert, R., J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, (2011). Natural language processing (almost) from scratch. The Journal of Machine Learning Research, 12, 2493-2537.

Damlijanovic, D., M. Agatonovic, H. Cunninng, (2012). The Semantic Web: ESWC workshops. Lecture notes in computer Science. Vol. 7177. 125-138.

Freitas, A., J. G. Oliveira, O' Riain, S. E. Curry, and J. C. P. Silva, (2011). Treo: best-effort natural language queries over linked data. In Natural Language Processing and Information Systems 286-289. Springer Berlin Heidelberg.

Giordani, A., and A. Moschitti, (2009). Semantic mapping between natural language questions and SQL queries via syntactic pairing. In Natural language processing and information systems 207-221. Springer Berlin Heidelberg.

Jurafsky, D. and H., Martin. (2000). Speech and Language Processing: An Introduction to Natural Languages Processing, Speech Recognition, and Computational Linguistics. Prentice-Hall, USA.

Kaur, K., and R. Rani, (2013, October). Modeling and querying data in NoSQL databases. In Big Data, 2013 IEEE International Conference on 1-7. IEEE.

Lehmann, J., and L. Bühmann, (2011). Auto Sparql: Let users query your knowledge base. In The Semantic Web: Research and Applications (pp. 63-79). Springer Berlin Heidelberg.

Li, Y., H. Yang, and H. V. Jagadish, (2005, June). NaLIX: an interactive natural language interface for querying XML. In Proceedings of the 2005 ACM SIGMOD international conference on Management of data 900-902. ACM.

Naeem, M. A., S. Ullah, and I. S. Bajwa, (2012). Interacting with data warehouse by using a natural language interface. In Natural Language Processing and Information Systems 372-377. Springer Berlin Heidelberg.

Norouzifard, M., S. H. Davarpanah, and M. H. Shenassa, (2008). Using natural language processing in order to create SQL queries. In Computer and Communication Engineering, 2008. ICCCE 2008. International Conference on 600-604. IEEE.

Orsi, G., L. Tanca, and E. Zimeo, (2011) Keyword Based Context Aware Selection of Natural Language query Pattrens 14th international conference on Extending Database Techonologies 189-200 ACM New York.

Sharef, N. M., and S. A. Noah, (2013). Natural language query translation for semantic search. International Journal of Digital Content Technology and its Applications, 7(13), 53-58.

Sordoni, A., Y. Bengio, H. Vahabi, C., Lioma, Grue J., Simonsen, and J. Y. Nie, (2015), October). A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (pp. 553-562). ACM.

Sujatha, B., V. Shaziya, and S. H. Raju, (2012).A Survey of Natural Languages interface to Database Management System. International Journal of science and Advance Technology, 2(6), 56-60.