ISSN No. 0976-5697

Volume 8, No. 5, May – June 2017

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

Review on conversion of natural language queries into SQL queries

Manisha
Department of Computer Science and Engineering
UIET, Kurukshetra University
Kurukshetra, India

Sona Malhotra
Department of Computer Science and Engineering
UIET, Kurukshetra University
Kurukshetra, India

Abstract: Natural language processing is a vast field of research in computer science which aims at simplifying interactions between computers and humans. The research creates an interface (NLIDB) which inputs the user query and then parses the query thereby generating an XML from that and then converts it into a SQL query that is finally executed to display the intended result. The input query is converted into sentence vectors and then filtering and stemming techniques are applied on it to finally generate a series of tokens which contain each word in its most raw format. This array of tokens is used to design conversion of query into a new language (XML based) i.e SQML, whose DTD will specify the tags which are permitted in language. We also create the query parser which creates the XML for the user query and finally, the XML is converted into SQL query. The research methodology will be programmed in java language due to its easy and quick XML generation and parsing. The conversion into SQML makes the query exportable into many other language independent formats like JSON and can be used for different purposes.

Keywords: Natural Language Interface to Database (NLIDB), Document type definition (DTD), SQL (Structured Query Language).

I. INTRODUCTION

Natural language processing is the emerging field of artificial intelligence in computer science in which there exists interaction between computers and humans which uses natural languages. Now a days, it is becoming one of the active areas in the interaction between humans and the computer. These include spoken language systems that integrate speech and natural language. This is an area of research which lies between linguistics and artificial intelligence, which is aiming to develop computer programs which are capable of human-like activities like understanding or producing texts or speech in a natural language, such as English or conversion of natural language in text or speech form into languages such as SQL[12]. Information retrieval and information organization, machine translation are major applications of natural language processing. NLP enables an interactive communication between persons and computers without a need to memorize commands of complex structures and procedures. A huge amount of labour is required if the required information is to be obtained from the entire repository of information system. In Natural language processing a user inputs his query in natural language (English language) which will then be converted into SQL query. Information retrieval from the database using natural language is a convenient way for those users who do not know about database query languages like SQL. Any ordinary person is not expected to know the SQL language, and hence this system would help him in generating the same, so that information retrieval is easier for the database, as database understand the SQL language only.

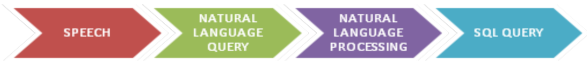

Figure 1. Demonstration of IRDBNLQ process

Assume that a business man want the details of top 10 customers from Delhi who have debited more than Rs.100000 from account .
So, SQL query should be like this

RDBMS> SELECT * FROM customers WHERE state='Delhi' and bal> 100000 SORT BY bal ASC LIMIT 10;

This is the conversion we expect our tool to perform in which the user inputs his query in natural language that is either in written or oral form, then conversion will be performed and desired result can be retrieved from the database.

II. RELATED WORK

Androutsopoulous et al. [1] in 1995 explained advantages and then, disadvantages of NLIDB, a comparison was made which was between NLIDB and formal query languages, graphical interfaces, form based interfaces. Linguistic problems were also discussed and then architecture was illustrated. Issues of portability, restriction of natural language input systems (which includes menu-based Nlidbs), and Nlidbs with reasoning capabilities were also discussed. There were some areas of Nlidb research which was less explored were then discussed and they are database updates, meta-knowledge questions, temporal questions, and multi-modal were also discussed and in the last the state of the art was explained.

Ana Maria Popescu et al. [2] in 2003, introduced a theoretical framework for reliable NLIs which was then the foundation for fully implemented precise NLI and researches proved that, for a large class of semantically tractable natural language questions, the proposed tool, PRECISE was guaranteed to map each natural language query to a SQL query. The author found the accuracy of the tool to be more than 80%.

Georgia Koutrica et al. [3] in 2010, various forms of structured queries were presented here as directed graphs and then the edges of the graphs were annotated with template labels which uses extensible template mechanism. Different graph traversal strategies were presented which efficiently explore the graphs and descriptions of textual query were also composed. Finally, he presented experimental results for the efficiency and effectiveness of the proposed methods.

Esther Kaufmann et al. [4] in 2010, four interfaces were introduced, each interface allow different query language. For all these interfaces benchmark was used to study its usability. A limited set of sentences were given in which keywords were given preference, showed in the result.

Lukas Blunschi et al. [5] in 2012, a system named as SODA (Search over Data Warehouse) were introduced. Design, implementation and experience with SODA were described. The gap which was developed between the need of business analysts and the complexities of current data warehouses were bridged by this system. For Google like experience keywords were taken from the queries entered by business users and an executable SQL were generated automatically by SODA.

In 2013, YingzhongXu et al. [6], QMapper was developed, so as to get optimized query that is HiveQL from SQL queries. Query rewriter and plan evaluator were its two main components. Rewriter was extended so as to support more rules and enhance the estimator to require the parallelism of multiple SQL blocks, data compression and the difference of cost effect between map and Reduce to provide a more fine-grained cost model.

Rakesh Kumar et al. [7] in 2014, performed a comparison between SQL and SQL on basis of their architecture, advantages, disadvantages and other features. A conclusion was made which conclude that the main reason for which the data was moving between SQL stores and Hadoop was taken so as to get an advantage of the massive storage and the processing capabilities so as to process large amount of data so as to cope with SQL.

Prasunkanti Ghosh et al. [8] in 2014, a tool was created which had integrated speech and natural language. Without a need to memorize the complex commands and procedures communication was enabled between non technical users and computers.

Bajwa et al. [9] in 2016, presented a natural language query based framework to generate SPARQL/RDF queries for NoSQL databases. In this research, a challenging task was to map natural language queries to map to SPARQL queries. This framework accepted input natural language query in English from user and then translated the NL queries into SPARQL/RDF queries. The results of the experiments with the designed framework reflected importance of such approach used for automated generation of SPARQL/RDF queries for NoSQL databases.

Song et al. in [10] in 2015, addressed these difficulties by developing a natural language-based system that allowed non-technical users to create well formed questions. A system named as TR Discover was proposed, in which the fragments of english which is a natural language were mapped into an representation method which is also known as First Order Logic representation, which was then either mapped into SPARQL or SQL. Mapping of natural language words into first order logic made it difficult to use featured based grammar which had formal semantics as well. The fragment of English covered by the natural language grammar was domain specific and tuned to the kinds of questions that the system could handle. Because users did not necessarily know what the coverage of the system was, TR Discover offered a novel auto-suggest mechanism that helped users to construct well-formed queries. TR Discover was developed to be used in future. Cortellis, was built that targets pharmaceutical domain and its user access it through keyword-based query interface. Results and the performance measures of TR Discover on Cortellis was reported, and in addition to that the portability of the system was demonstrated on QALD-4 dataset, which was associated with a public shared task. The portability and usability of the system was showed and relative performance of the queries was reported by the user as SQL and SPARQL was used at back end.

Yaghmazadeh et al. [11] in 2017, a technique was proposed in which SQL queries were synthesized from natural language queries automatically. Fully automated technique was build, which can work for any database without requiring additional customization, and did not require users to know the underlying database schema. The proposed method had achieved its goals by combining natural language processing, program synthesis, and automated program repair. The user's description in english was given, firstly semantic parsing was used to generate query sketch, which was then be completed by using type directed program synthesis and a confidence score was assigned using the content of the database. However, since the user's description did not accurately reject the actual database schema, this approach also performed fault localization and repaired the erroneous part of sketch. The synthesize-repair loop was repeated until a query was inferred which had high confidence score. Sqlizer was the tool in which the technique was implemented and evaluation was done on three different databases. The experiments showed that the desired query was ranked within the top 5 candidates in close to 90% of the cases.

III. PROBLEM FORMULATION

To retrieve data from relational databases are considered to be challenging. Standard query language (SQL) is expressive and powerful for relational databases but, is difficult for those users who are not technically sound. As even for those users who have expertise in database programming languages find it difficult because it require users to know the exact schema of the database. Now, userbase is shifting towards non-experts, who are responsible for designing user-friendly query interfaces. So, to fulfill the needs of those non-technical users, an interface is needed which is convenient for them as well.

IV. METHODOLOGY

Retrieving information from databases is always considered to be job of a database expert who has knowledge of SQL to interact with the database. Now, interacting with a database through a natural language statement requires identification of proper table names, fields, constraints or mixed expressions. This method needs to map each word in user statement to be mapped to a database term. To do this, first step is to get the words to their basic form through stemming algorithms. These words along with their synonyms are then mapped. SQML derived from XML. DTD (Document type definition) is to be defined for our SQML which declares the tags, their multiplicity and their validity to understand the XML tags defined therein. The steps are as follows:

Step 1: Input the natural language query.
Step 2: Split the input query to form an array of vectors where each vector corresponds to a statement in the input query.
Step 3: Perform filtering to remove words like is, am, are etc.
Step 4: Then stemming is performed to get the words to their raw form for better comparison.
Step 5: Now each word in the lists is mapped to a database specific word and this mapping also considers the synonyms of each noun involved there.
Step 6: These mapped words are then fitted into a XML file to form an XML entity which represents the present query.
Step 7: Now a conversion algorithm is to be defined to finally convert this XML into a SQL query.
Step 8: The query is then executed and the results obtained are displayed to user.

V. CONCLUSION AND FUTURE SCOPE

Here, a novel technique is defined to convert the natural language queries into SQL queries. The research creates an interface (NLIDB) which inputs the user query and then parses the query thereby generating an XML from that and then converts it into a SQL query that is finally executed to display the intended result. The conversion into SQML makes the query exportable into many other language independent formats like JSON and can be used for different purposes. Conversion to XML is always a better method than previous techniques where the queries were converted into graphs and then graphs into SQL queries. This is because XML parsing is far more easier and efficient than graph traversal. So the introduction of QML will certainly enhance the performance.

In future, methods can be defined to export QML into other commonly used formats to increase the acceptability of the language. A limited data dictionary is to be used and this dictionary needs to be updated regularly with those words that are specific to some particular system. Here, the question string will be splitted into tokens and order number will be provided to each token. To remove excessive words from the user input statement. Escape words have been considered which must be regularly updated with words that are specific to the particular system. To construct RDBMSQL query using tokens, an algorithm has to be devised. Ambiguity among the words will be taken care of while processing the natural language.

VI. REFERENCES

[1] Androutsopoulous, Ritchie and Thanischin, *"Natural language interfaces to databases-An introduction"* Arxiv, Journal of natural language processing, Cambridge university press, Vol. 2 P. No. 709, 1995.

[2] Ana Maria, Oren Etzioni and Henry Kautz in "*Towards a theory of natural language interfaces to databases"*, Springer ACM Pg. 586-594, 2003.

[3] Georgia Koutrica, AlkisSimitsis and Yannis E. Loannidis in *"Explaining structured queries in natural language"*, IEEE, 2010.

[4] Esther Kauffmann and Abraham Bernstein in *"Evaluating the Usability of Natural Language Query Languages and Interfaces to Semantic Web Knowledge Bases"*, Journal of web semantics, 2010.

[5] Lukas Blunschi, Claudio Jossen and Donald Kossman in *"SODA: generating SQL for business users"*, International conference on very large databases, Vol. 5 No. 10, 2012.

[6] YingzhongXu and Songlin Hu, *"QMapper: A Tool for SQL Optimization on RDBMS Using Query Rewriting"*, ACM WWW pp. 211-212, 2013.

[7] Rakesh Kumar, Neha Gupta, Shilpi Charu, Somya Bansal and Kusum Yadav, "*Comparison of SQL with SQL*", International Journal for Research in Technological Studies, Vol. 1, Issue 9, pp. 28-30, 2014.

[8] PrasunKantiGhosh, SaparjaDey and Subharta Sengupta in *"Automatic SQL formation from Natural Language Query"* International conference on micro-electronics, circuits and systems, 2014.

[9] I. S. Bajwa, F. Razzaq, A. H. S. Bukhari and R. Amin, *"Using Machine Learning to Generate SPARQL Queries from NL for NoSQL Database"*, Sindh Univ. Res. Jour. (Sci. Ser.) Vol. 48, No. 2, pp. 233-240, 2016.

[10] Dezhao Song, Frank Schilder, Charese Smiley, Chris Brew, Tom Zielund, Hiroko Bretz, Robert Martin, Chris Dale, John Duprey, Tim Miller and Johanna Harrison, *"TR Discover: A Natural Language Interface for Querying and Analyzing Interlinked Datasets"*, Springer, pp. 21–37, 2015.

[11] Navid Yaghmazadeh, Yuepeng Wang, Isil Dillig and Thomas Dillig, "*Type and Content Driven Synthesis of SQL Queries from Natural Language*", 2017.

[12] Cheshta and Tarun Bagga, "*Retreiving unstructured data from RDBMS using natural language queries*", Vol. 5, Issue 4, 2015.