Natural Language Interface to Database
Mahesh P. Gaikwad

Abstract—The need for natural language interfaces to database has become increasingly acute as more and more people access information from web browsers. Yet NLI (Natural Language Interface) is only usable if they map natural language questions to SQL queries correctly. Natural Language processing is becoming one of the most active areas in Human-Computer Interaction. It is a branch of AI (Artificial Intelligence) which includes in information retrieval, machine translation and Language analysis. The goal of NLP (Natural Language Processing) is to enable communication between people and computers without requiring to memorization of complex commands and procedures. In other words NLP (Natural Language Processing) is techniques which can make the computer understand the languages naturally used by humans. The main purpose of natural Language Query processing is for an English sentence to be interpreted by the computer and appropriate action taken. Asking questions to databases in natural language is a very convenient and easy method of data access, especially for casual users who do not understand complicated database query languages such as SQL.

Keywords: NLP, Natural Language Interface to Database.

I. INTRODUCTION

NLIDB (Natural Language Interface to Database) is a technique which can make the computer understand the languages naturally used by humans, but not by artificial or man-made language such as a programming language. The main aim of the topic is asking questions to databases in natural language which is a very convenient and easy method of data access, especially for casual users who do not understand complicated database query languages such as SQL [2].

- To allow easy access to database.
- Nontechnical person can also access database without using SQL queries.
- Translate the text into another language.
- Answer questions about the contents of the text.
- Use of database is world-wide today so NLIDB can allow easy interface for users.
- Allows user to interact with database system without knowing database schema.
- To work with any RDBMS one should know the syntax of the commands of that particular database software (Microsoft SQL, Oracle, etc.),
- Here the natural language processing is done on English i.e. the input statements have to be in English.
- Input from the user is taken in the form of questions (WH - words like what, who, where, etc).

II. LITERATURE REVIEW

One area of research efforts in the query interfaces is focused on improving the usability. The main goal is to provide a high level interface that can be used by nontechnical users without any requested DBMS oriented knowledge. An important area in this direction is the application of natural language interface for databases (NLIDB)[1]. The NLIDB means that a user can use some natural language to create query expressions and also the answer is presented in the same language. The history of NLIDB goes back as early as 1960’s. The era of peak research activity on NLIDB was in the 1980’s. In that time, the development of a domain and language independent NLIDB module seemed as a realistic task. The prototype projects showed that the building of a natural language interface is a much more complex task than it was expected. Regarding the usability of NLIDB, there can be found some tests in the literature that evaluates the efficiency of the NLI interfaces. In these tests the NLIDB is compared with traditional interfaces like SQL [3]. The results show that expert users can perform more efficiently the special command interface (SQL) than the NLI interface. On the other hand, the un-experienced users could achieve better results with the NLI interface than with the imperative SQL interface.

A. Critical appraisal of other people:

The people who watched the working of our project, they find it is the great invention in the Natural Language Processing Field. The projects demonstrate the idea behind the Processing of query without having any Technical Knowledge.

B. Investigation of current Project and Related work:

The very first attempts at NLP (Natural Language Processing) database interfaces are just as old as any other NLP research. In fact database NLP may be one of the most important successes in NLP since it began. Asking questions to databases in natural language is a very convenient and easy method of data access. Especially for casual users who do not understand complicated database query languages such as SQL. The success in this area is partly because of the real-world benefits that can come from database NLP systems, and partly because NLP works very well in a single-database domain. Databases usually provide small enough domains so that ambiguity problems in natural language can be resolved successfully. Here are some examples of database NLP systems:

- LUNAR (Woods. 1973) involved a system that answered questions about rock samples brought back from the moon. Two databases were used, the chemical analyses and the literature references. The program used an Augmented Transition Network (ATN) parser and Woods Procedural Semantics. The system was informally demonstrated at the Second Annual Lunar Science Conference in 1971.
- LIFER/LADDER was one of the first good database NLP systems. It was designed as a natural language interface to a database of information about US Navy ships. This system, as described in a paper by Hendrix (1978), used a semantic
grammar to parse questions and query a distributed database [5]. The LIIFER/LADDER system could only support simple one-table queries or multiple table queries with easy join conditions.

III. ARCHITECTURE

![Architecture Diagram]

IV. ALGORITHMS

A. Tokenizer:
I/P: Inputted Question.
O/p: Separated Tokens from inputted question.
1. Start (Input Question).
3. Check the syntax of question (Parse inputted question)
   a. Parse (String S)
   b. If the question contains special characters, extra spaces then remove them.
   c. If the question start with ‘wh’ word command like question then syntax is correct.
   d. Split the question in tokens ‘S’
   e. Else report Invalid Question Format Message
4. If syntax is correct then split query in tokens & remove the syntactic markers
   a. If tokens[S] contains Syntactic markers then remove Syntactic markers
   b. Else Display the Separated tokens.
5. Display separated Tokens
6. End

B. Matcher:
I/P: Separated Tokens
O/P: Table name and Attribute names.
1. Accept the generated tokens in String ‘W’.
2. Apply Tokenizer (String W).
   a. For all Tokens W[i]
5. Display separated Tokens
6. End

C. SQL Translation:
I/P: Table_name, Column_names, Value_set.
O/P: Sql query.
1. Accept the Table_name,Column_names,Value_set
2. Map the Table_name and Column_names to SQL query format: Query_generator (
   a. Place column_names and Table_name at proper position in sql query.
   b. If value_set found then
5. Display Result:
I/P: Table_name, Column_names, Value_set.
O/P: Sql Query.
1. Get the generated SQL query in String str.
2. Fire the SQL query on Database: GetTableContent (str).
3. Get the Result in array and display in table.

V. CONCLUSION
This system can used to retrieve the data from large databases like College Administration System, Railway reservation and enquiry machine, Customer care services, Computerised and Online Dictionaries.

REFERENCES

ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 8, February 2013

[3] Fred Popowich, Simon Fraser University “Using Text Mining and Natural Language Processing for Health Care Claims Processing”.

[4] John Pestian Nanyang Technological University, 639798 Singapore “Neurolinguistic Approach to Natural Language Processing with Applications to Medical Text Analysis”.


