Natural Language Processor Statistical Processing

Akshay Ghaisas[1], Rishikesh Jadhav[2], Preyas Nandedkar[3], Nikhil Narkhede[4]

[1]*Student Department of Computer Engineering MMCOE, Pune*

[2]*Student Department of Computer Engineering MMCOE, Pune*

[3]*Student Department of Computer Engineering MMCOE, Pune*

[4]*Student Department of Computer Engineering MMCOE, Pune*

Marathwada Mitra Mandal's College of Engineering,Pune-52

***Abstract*— The field of natural language processing (NLP) has seen a dramatic shift in both research direction and methodology in the past several years. In the past, most work in computational linguistics tended to focus on purely symbolic methods. Recently, more and more work is shifting toward hybrid methods that combine new empirical corpus-based methods, including the use of probabilistic and information theoretic techniques, with traditional symbolic methods. 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. This paper proposes the architecture for translating English Query into SQL using Semantic Grammar.**

I. INTRODUCTION

Natural language processing is a theoretically motivated range of computational techniques for analysing and representing naturally occurring texts at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of tasks or applications. The general goal for most computational linguists is to instill the computer with the ability to understand and generate natural language so that eventually people can address their computers through text as though they were addressing another person. The applications that will be possible when NLP capabilities are fully realized are impressive computers would be able to process natural language, translating languages accurately and in real time, orextracting and summarizing information from a variety of data sources, depending on the users' requests. Our natural language processor is based on statistical processing approach. Statistical approaches employ various mathematical techniques and often use large text corpora to develop approximate generalized models of linguistic phenomena based on actual examples of these phenomena provided by the text corpora without adding significant linguistic or world knowledge. The project is being implemented in django/python platform. The database is built in postgres which is an open source. After successful login user will place his query in natural language e.g. English. This query will be processed and converted into intermediate query. In next phase this intermediate query will get converted into actual postgres query. In last phase results will be fetched and displayed in the form of charts and tables. These results will be validated by the user and appropriate feedback will be taken. using these feedback results will be optimized.

II. REALATED WORK

The very first attempts at NLP database interfaces are just as old as any other NLP research. In fact

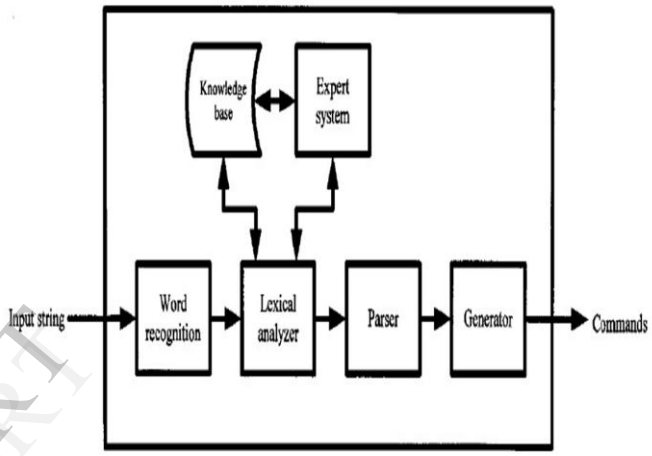
database NLP may be one of the most important successes in NLP since it began. Asking questions to databases in natural language is a very convenient and easy method of data access, especially for casual users who do not understand complicated database query languages such as SQL. The success in this 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 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. The LIFER/LADDER system could only support simple one-table queries or multiple table queries with easy join conditions.

III. SYSTEM DESCRIPTION

A brief description of the system is as follows:

Suppose we consider a database say POSTGRES.Within this POSTGRES database I have placed certain tables, which are properly normalized. Now if the user wishes to access the data from the table, he/she has to be technically proficient in the SQL language to make a query for the POSTGRES database. Our system eliminates this part and enables the end user to access the tables in his/her language. Let us take an example: Suppose if we want to view information of a particular threat from VIRUS table then we are supposed to use the following query: SELECT * FROM VIRUS WHERE v_name ='WX23'; But a person, who doesn't know POSTGRES, will not be able to access the database unless he/she knows the syntax and semantics of firing a query to the database. But using NLP, this task of accessing the database will be much simpler. So the above query will be rewritten using NLP as: Give the information of viruses whose name is WX23. Both the SQL statement and NLP statement to access the VIRUS table would result in the same output the only difference being, a normal person who doesn't know anything about SQL can easily access the POSTGRES database.



IV. SYSTEM SCOPE

The search engine that we are building is basically for the web portal of Symantec Corporation. The search results that will be displayed are going to be statistical data. E.g. if the input string in the search is "what are the top 10 Trojans in the last week" then it will display the statistical analysis of 10 Trojans that were found in last week. Here the key statistical terms are "10 Trojans" and "week".

The results of the search will be displayed in the form of tables, charts and graphs. The user will give input in simple English language which will be converted into the database query in the further phases. In the first version it is restricted to English language but in further extension we are planning to include other languages too.

The main feature of this application is that it is pluggable. By pluggable we mean that it is not going to be restricted to a single database but can be applied to any database. This unique feature will provide an edge over currently available applications of this domain.

The search algorithms are based on tree data structure which is generally static but we are aiming to make it dynamic. By this we mean that for every search criteria the tree structure will change dynamically according to the feedback from the user. To get the feedback from the user we are going to give him the options regarding the search result. The options will be: correct result, incorrect result.

V. SYSTEM REQUIREMENTS

- Hardware Requirements
  - •Processor-Pentium 4 or higher version
  - •RAM – 128Mb(256 Recommended)
  - •HDD- 20 GB (40 GB Recommended)
  - •Network Adapter
  - •Internet Connection(minimum 40 KBPS speed)

- Software Requirements
  - •Python
  - •Postgres
  - •Open Source OS/Window OS(recommended)
  - •Any web browser

VI. SYSTEM ARCHITECTURE

• Front End Module: Will basically contain GUI. Here user will place his query in natural language e.g. English.

• Natural Language Processing routine: Python platform will be used to design this module. In this module parsing of the input string will be done. This phase will generate statistics related tokens which will be passed to the query generator.

• Query Generator: With the help of the tokens received from the parser in the previous phase this subroutine will generate an appropriate query.

• Back End Module: There will be basically two types of databases.

- Static – Vocabulary database
- Dynamic – Information database

VII. ALGORITHM

1. Scan the query.
2. Tokenize.
3. Compare tokens with stat_table(Table containing all statistical words) & insert in Table temp_buff(A table used for intermediate query) if found
4. Else Compare tokens with table with qnoun_table (this table contains all table names with their synonyms having same type) if found take its type and find its actual table name from meta_table(A table which contains all the names of the tables of the threats database)
5. Using type and actual table name insert it in temp_buffer with flag=1
6. Else discard
7. Scan Flag field of the table temp_buff, if all flag vales are equal to zero then discard the whole query.
8. From this temp_buff table generate actual POSTGRES query using map table (this table contains necessary part of POSTGRES query which is required to built whole query).

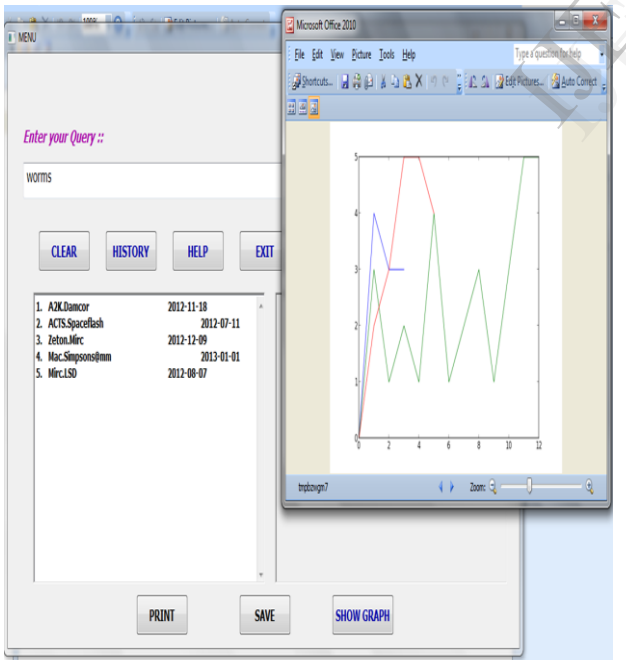
VIII. RESULT

The graphical User Interface is as shown below. Initially when installing the system, the user needs to provide the username, password, IP address. Once the setup is done the user can start the application and perform the following steps to get the result.

Steps followed to get the result:-

- Type the natural language query in the dialogue box given.
- Click on “Go” button.
- Then the result will be displayed.

- The system will ask the user for the feedback about the result.
- If the user feedback is negative regarding the result displayed then after submitting the feedback that query will be sent by e-mail to the admin.



IX. CONCLUSION

Natural Language Processing can bring powerful enhancements to virtually any computer program interface. This system is currently capable of handling simple queries with standard join conditions. Because not all forms of SQL queries are supported, further development would be required before the system can be used within NLDBI. Alternatives for integrating a database NLP component into the NLDBI were considered and assessed.

X. REFERENCES

• Allen J., Natural Language Understanding, Benjamin/Cummings, Redwood City, CA, 1987.
• Davis, W. S., Computers and Information Systems: An Introduction, West, Minneapolis, MN, 1997.
• Harald Baayen. Analyzing Linguistic Data: A Practical Introduction to Statistics Using R. Cambridge University Press, 2008.
• Nidhi Mishra. Part of Speech Tagging for Hindi Corpus, 2011 International Conference on Communication Systems and Network Technologies.
• Thales Sehn Korting. C4.5 algorithm and Multivariate Decision Trees,2008.
• Rekha S. Sugandhi, Ritika Shekhar. Issues in Parsing for Machine Aided Translation, 2011 IEEE
• Samar Husain, Phani Gadde, A modular cascaded approach to complete parsing, 2009 International Conference on Asian Languages Processing

Address for correspondence:
FLAT NO. 2, RAJ RESIDENCY, ANAND NAGAR, SINHAGAD ROAD, PUNE 411051