Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 7, Issue. 11, November 2018, pg.1 – 7

NATURAL LANGUAGE PROCESSING IN COOPERATIVE QUERY ANSWERING DATABASES (NLPiCQA)

Edmund Sowah

Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China
esowah2000@gmail.com; esowah2000@nuaa.edu.gh

Abstract - It has been the desire of casual users of databases and information system to have systems that minimize or eliminate the communication gap between computers and humans. Even though, in recent times, almost all computer and mobile applications store information in databases, the process of getting access to this information has not been easier for casual users. Retrieving information from databases requires technical proficiency in Structured Query Language (SQL) and knowledge of the database structure and tuple attributes. The aim of this paper is present a cooperative approach to information search and retrieval in databases and information systems. This paper present an architecture that combines Natural Language Processing (NLP) and Cooperative Query Processing (CQA) to develop a cooperative database system. The architecture proposed enables casual users to submit queries in plain English language and the system returns cooperative results – results more than what the user will normally require.

Keywords - Natural Language Processing, Cooperative Query Answering, Structured Query Language, Cooperative database system, information retrieval

I. INTRODUCTION

Databases are comprehensive element in private and public information systems, which are essential in number of application areas [1]. Traditional database systems have been of great importance for a long time. In as much as they are still important, they have not been easier to be used by casual users. There are many information stored in database systems but getting access to them is rather difficult to casual computer users. This is because users need to submit queries in SQL syntax in order to get information from traditional database systems. In addition to writing

SQL syntax, casual users need to know the database structure and table or tuple attributes to be able compose "perfect" queries to gain detail information.

It is no surprise that with the advancement in Artificial Intelligence (AI), most people are demanding for a much easier approach in accessing information in databases and information systems. Casual users prefer asking questions (submitting queries) without bothering about the database structure and SQL syntax. Casual users prefer to compose and submit their queries in natural languages. This is because asking questions using natural language to get results from databases is a very convenient and easy method of data access [2, 3]. The act of talking to a computer in a natural language such as plain English is always a dream that drives the progress of human-computer interaction work [4]. Users also require databases and information systems to provide information more than what the query would normally generate. Even though the need for a more intelligent database system with the capacity of understanding and responding to natural language queries has been addressed in several researches [5] , these two demands from casual users can be achieved by combining two methodologies – Natural Language Processing (NLP) and Cooperative Query Answering (CQA).

Natural Language Processing (NLP) is an area of Artificial Intelligence (AI) that requires computer systems to process natural language. The central objective of NLP is to create friendly, computer interaction environment that does not require skills in computer programming language. Casual users of database systems do not possess the necessary skills and knowledge to formulate and understand database query languages like SQL (Structured Query Language). Thus, the need to make it possible for users to send queries in natural language is very important in this current dispensation of database development and usage. Database NLP provides the "real-world benefits" of using database systems, and works efficiently in single-database domains [6].

Cooperative Query Answering (CQA) is the method of providing results to queries in a human manner. By this method, the database system should act as closely, to how humans provide response to questions. Cooperative database system provides users with an intelligent database interface that allows them to issue approximate queries independent to the underlying database structure and its content, and provides additional useful information as the exact answers [1].

II. RELATED WORK

In most databases and information systems, cooperation with the users is uncommon. Users of databases and information systems need to know the underlying database structure and compose an accurate and in-depth SQL queries in other to generate direct answers. The need to develop databases and information systems that can predict and accept users' queries in simplified natural language and SQL has led to many findings and development. The well-known works concerned with NLP and cooperative systems are as follows:

- *CARMIN* [7] was developed as a research platform for cooperative answering and as a practical and efficient Cooperative Database System (CDBS) in the University of Maryland. Carmin was developed using Datalog and INGRES. Users of the system submitted queries in Datalog through INGRES front-end user interface. The system analyses query, translate the query into SQL and migrates the translated query into a Relational Database Management System (RDBMS) for evaluation.

- *CoBase* [8] uses Type Abstraction Hierarchy (TAH) as a framework to generate cooperative Query Answers (CQA). The CoBase project introduced a conventional query language – CSQL, which is a cooperative extension of SQL to enhance query processes and evaluation. The project was developed and tested in University of California, Los Angeles but was applicable in areas such as medical image systems (KMed), transportation logistic planning (GLAD) and noisy emitter identification systems.

- *LIFER/LADDER* [9] has been one of the significant application of natural language processing in database systems. LADDER allows users to submit questions about the information base in natural language (English) and uses developed components to transform these questions. The researchers developed LIFER

(for language interface facility with ellipsis and recursion) to process natural language questions as part of the LADDER project. LIFER/LADDER processes one-table queries and multiple queries with easy join conditions.

III. SYSTEM DESCRIPTION

Natural Language Processing enhances human-computer interactions and provide a means through which casual users of database systems can get useful information. Cooperative query answering makes database systems more useful to users than traditional database systems, which provides direct answers. A cooperative result to a query is an indirect response that is more helpful to the user than the direct response would be. The goal of this paper is to produce an architecture of a Cooperative Query Answering database system with Natural Language Processing (NLP) capability. This project will further improve communication between database users and database systems in a natural way over traditional way of database interactions.

The increasing number of casual users of database system has made it necessary to build databases and information systems that accepts queries in plain English (natural language) and provide information more than what the user will normally require.

In order to understand the architecture this paper has developed, let us consider a Flight Schedule Database, which has been created using MySQL for an Airport. Within the Flight Schedule database, there is a table (Table 1), that is normalized. If a casual user wishes to access information from the database, it is advisable and mandated that, the user is technically capable of writing queries in SQL. The user should also have knowledge about the database structure and table properties to be able to formulate and submit queries that can generate useful information. The architecture and methodology proposed in this system eliminates these challenges, but instead, enables the user to submit queries in natural language.

TABLE I
Flight Table

Flight	Dept_Time	Dept_City	Arr_City	Work_Days	Price
NJ008	500 hrs.	Nanjing	Shanghai	Monday, Friday	120
BJ109	700 hrs.	Nanjing	Beijing	Tuesday	80
NJ055	1900 hrs.	Nanjing	Beijing	Wednesday	110
SG044	2000 hrs.	Nanjing	Shanghai	Thursday, Sunday	150
WX889	1200 hrs.	Wuxi	Harbin	Monday, Saturday, Sunday	50
NJ468	1600 hrs.	Nanjing	Wuxi	Tuesday, Thursday	45
NT671	1100 hrs.	Wuxi	Shanghai	Sunday	70

Let us take an example: Suppose a user want to view information such as all flights going to Shanghai from Nanjing on any day from the Flight Schedule database. The user will have to formulate and submit the following SQL statement as a query:

SELECT flight, Arr_City, Work_Days FROM Flight WHERE Arr_City = "Shanghai".

It is important to stress that a user without knowledge of SQL syntax and structural knowledge of the flight schedule database cannot write such a query. Contrary, by using NLP, the user will find a much easier means of interacting with the database. The above SQL syntax can be rewritten using NLP as a question in cooperative database system interface as:

What are the flights going to Shanghai from Nanjing on any day?

Both the SQL statement and NLP statement will produce the same result in traditional database systems, even though NLP makes the process of query submission easier. Table 2 depicts the results from a traditional database system.

TABLE II
Result from Traditional Database System

Flight	Arr_City	Work_Days
NJ008	Shanghai	Monday, Friday
SG044	Shanghai	Thursday, Sunday

The above queries will produce quiet different result in a cooperative database. As explained, cooperative query answering provides more results (information) than what is produced in traditional database systems. From Table 3, it can be seen that the query produced three results instead of two, as in the case of a traditional database system. The result includes a flight departing from Wuxi (a city close to Nanjing) to Shanghai. This addition gives extra information to the user and improves his/her flight choices.

TABLE III
Result from Cooperative Database System

Flight	Arr_City	Work_Days
NJ008	Shanghai	Monday, Friday
SG044	Shanghai	Thursday, Sunday
NT671	Shanghai	Sunday

IV. SCOPE OF THE SYSTEM ARCHITECTURE

The scope of the proposed system (NLPiCQA)is as follows:

- The core database structure is a Relational Database Management System (RDBMS).
- English is the only natural language acceptable.
- The type of queries acceptable are imperative and declarative queries [10].
- Input from users can be in question forms, statements and keywords.
- There are no “danger” words. Users can write as many words as necessary. Grammatical and spelling errors are not rejected but will lead to less detailed information.
- To remove excessive words from user input statements, a stopwords list has been created.

V. SYSTEM ARCHITECTURE

The proposed system architecture is concerned, first, with accepting user queries in natural language. The system will enable users to compose queries using plain English language. Users do not need skills and knowledge in SQL and database structure. Secondly, the system will be able to transform the user queries (in plain English) to corresponding SQL query within the database system. Finally, the system after processing the query will generate a cooperative result to the users.

Fig. 1 depicts the system architecture of NLPiCQA. The system developed includes the following components:

- *Graphical User Interface:* This allows the user to submit queries in natural language to the system.
- *Tokenizer:* This splits the query into individual words and orders them numerically.
- *Keywords Extractor:* The tokenized words are checked against the words in the stopwords list. Excessive words are removed and keywords are extracted.
- *SQL Query Generator:* The key words are used to formulate SQL queries.
- *Query Executor:* The query is executed within the database system.
- *Result Collection and Display:* Result generated from executing the query are stored and formatted according to designed format and displayed to the user at the front-end.

TABLE IV
List of MySQL Default Stopwords

a	about	an	are	as	at	be	by
com	de	en	for	from	how	i	that
in	is	it	la	of	on	or	where
who	the	this	to	was	when	what	will
with	the	und					

VI. QUERY TRANSFORMATION AND EXECUTION ALGORITHM

The following steps explains the algorithm used in query transformation and execution in the NLPiCQA.

- Receive user query input
- Tokenize user queries
- Check the tokenized words against stopwords list
- Extract keywords
- Use keywords to formulate SQL statement
- Execute the SQL query.
- Result set generated
- Result is collected, formatted and organized
- Display result to user.

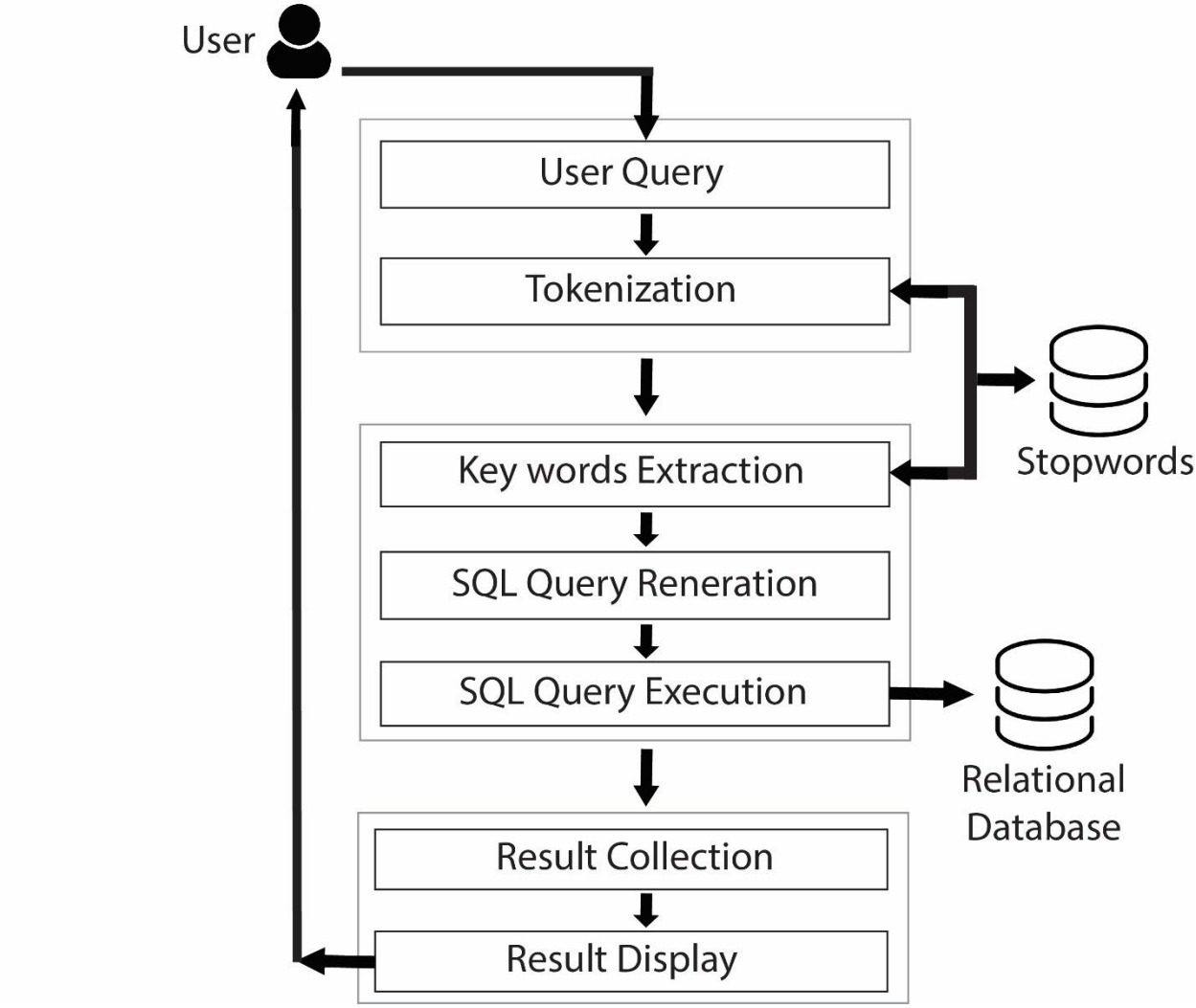

Fig. 1 Architecture of NLPiCQA

VII. CONCLUSION

Combining Natural Language Processing (NLP) and Cooperative Query Answering (CQA) to develop a database system results in a powerful and efficient system that enhances and improves human-computer interactions. The architecture proposed in this paper shows the possibility of combining the features of these methodologies to produce an efficient database system. This paper demonstrated how users can submit queries to cooperative database systems using natural language and receive information beyond what is normally obtained from traditional database systems. The architecture developed is a step above what is currently used in cooperative query answering database. The proposed architecture, unlike the rest do not restrict the user to word-choices and syntax. This approach bridges the gap between humans (casual users) and computer systems.

Even though this paper did not put the architecture in practical use that remains our future development. In the future, we will apply the architecture developed in addition to other system development application or programming language. The overall future development aim is making it practical for casual users to use databases and information systems with their own language.

REFERENCES

[1] R. Halder and A. Cortesi, Cooperative Query Answering by Abstract Interpretation. *International Conference on Current Trends in Theory and Practice of Computer Science SOFSEM* 2011: *Theory and Practice of Computer Science*, pp 284-296.

[2] I. Androutsopoulos, I., G.D. Ritchie, and P. Thanisch, Natural Language Interfaces to Databases – An Introduction. *Natural Language Engineering*, 1995. 1(1):29-81.

[3] F. Siasar djahantighi, et al., Using Natural Language Processing in Order to Create SQL Queries, *Proceedings of the International Conference on Computer and Communication Engineering* 2008, Kuala Lumpur, Malaysia.

[4] M. Brodie, "Future Intelligent Information Systems: AI and Database Technologies Working Together" in *Readings in Artificial Intelligence and Databases*. Morgan Kaufman, San Mateo, CA, 1988.

[5] D. P. Mckay, and T. W. Finin, "The Intelligent Database Interface: Integrating AI and Database Systems", *In Proceedings of 1990 National Conference on Artificial Intelligence*, pp. 677-684, 1990.

[6] A. Rukshan, P. Rukshan and S. Mahesan, Natural Language Web Interface for Database (NLWIDB). *Proceedings of the Third International Symposium, SEUSL*: 6-7 July 2013, Oluvil, Sri Lanka.

[7] P. Godfrey, J. Minker and L. Novik, An Architecture for a Cooperative Database System. *International Conference on Applications of Databases, ADB* 1994: *Applications of Databases* pp. 3-24.

[8] W. W. Chu, Q. Chen and M. Merzbacher, CoBase: A Cooperative Database System in Non-Standard Queries and Answers. *R. Demolombe and T. Imielinski* ed., 1194.

[9] G. G. Hendrix, et al., Developing a natural language interface to complex data, in *ACM Transactions on database systems*, 1978, 3(2), pp. 105- 147, 1978.

[10] M. De Calmes, D. Dubois, E. Hullermeier, H. Prade and F. Sedes, Flexibility and fuzzy case-based evaluation in querying: An illustration in an experimental setting. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2003, 11 (1), pp. 43-46.

