An Advanced Web-Based Hindi Language Interface to Database Using Machine Learning Approach

Zorawar Singh Virk(✉) and Mohit Dua

Department of Computer Engineering,
National Institute of Technology, Kurukshetra, India
zorawarvirk@yahoo.ca, er.mohitdua@gmail.com

Abstract. NLIDB (Natural Language Interface to Database) system is a key step to many areas like Database Mining, Search Engines, Medical Science, Artificial Intelligence etc. Existing literature reveals that Natural Language Interfaces were extensively studied from the 1970's to 1990's. The paper proposes a NLIDB system namely Advanced WB-HLIDB (Web based Hindi Language Interface to Database using Machine Learning Approach) that uses the natural language as Hindi language and is based on the Machine Learning technique. The main features that have been introduced are Web Based Graphical User Interface to the system along with the use of Clustering and Similarity functions. This paper discusses the use of Similarity approach instead of the 'Like' approach which had been in practice till recently. Various other components such as multiword selection, recognizing misspelled words, storing of successful queries, auto complete function, built in keywords etc. have also been implemented.

Keywords: NLIDB · WB-HLIDB · HLIDB · Machine learning · Data mining · Similarity functions · KNN algorithm · Hindi shallow parser

1 Introduction

As there is a volatile growth in the field of Internet, the need of the hour for organizations that provide services like information storage, access as well as the information analysis is also increasing at an amazingly high rate. These days, there is too much of the data that needs to be maintained in organizations, companies and university's databases. Only the individuals who are familiar with formal query languages such as SQL, can directly use or access this data. Thus the primary motivation behind studying and implementing a Natural Language Interface to Database System is to allow the user to access as well as maintain the database in the form of natural language query and extend this to any given database in general. Information in a NLIDB system is mainly stored in a structured manner in the form of tables. Thus giving the user an option to query the database by asking questions in natural language instead of a query language like SQL (Structured Query Language) can be of great convenience since the user need not remember the syntax for various queries. Here, Natural Language refers to the typical or the common language that is widely spoken and understood like English or Hindi.

S. Yamamoto (Ed.): HIMI 2016, Part I, LNCS 9734, pp. 381–390, 2016.
DOI: 10.1007/978-3-319-40349-6_36

Similarity measures have become an extremely popular tool in machine learning. One of the biggest problem that haunts the NLIDB system is that of Data Mining. Data Mining refers to the manner in which the useful entities are extracted or retrieved accurately and efficiently from a database containing large volumes of raw data. This is done using a process known as approximate data matching process which further relies on the concept of different similarity functions. The concept of similarity can be different depending on particular domain, task or dataset available. It is desirable to learn similarity functions from training data to seize the correct notion of distance for a particular task available in a given domain.

1.1 NLIDB Areas

Various methods as well as software have been designed so as to solve this problem, but these methods or software do not have the ability to handle complex queries. In the field of Medical Science, NLIDB systems can enable the patient or any general concerned user to access information related to any medical disease or illness, medicines, preventions and precautions etc. This is extremely useful in areas where those who provide information are far less than those who need information.

Another area where NLIDB systems can prove to be immensely convenient is Railway enquiry. In a country like India, which has one of the world's largest railway networks, there is tons of information regarding time tables of thousands of trains running on different routes across the country. Thus having a NLIDB System which can handle all user queries related to the railways can be really beneficial. This type of system simplifies the process of data retrieval as well as data management from databases without making it mandatory for the users to have any prior knowledge regarding the formal query language syntax.

Hence, NLIDB is very appropriate solution for users to express their queries. But as the internet availability as well as the need for these kind of systems increases another aspect of the hour that needs to be addressed is to increase the scope as well as the portability of these type of systems.

2 Related Work

Research in the area of Natural Language Interfaces (NLIs) has been started from the early fifties. From the end-user point of view natural language is easy to use as it is used every day in human to human communication, and is therefore considered as a useful and efficient way for people to interact with computers. Versions of NLIDB had come in the late 60's and early 70's. A number of NLIDB systems have been developed since. Lunar [1] system was built to answer the queries regarding the samples brought back from the moon.

SQ-HAL [2] system was designed to have multiuser support as well as being database and platform independent. Khalid, M.A [3] described QA system which used the information extraction module to make the training data of classifier and this system was designed for English language. HLIDB [4] discussed the various issues, challenges

and techniques involved in building an NLIDB. The article also proposes an approach to reduce costs involved in system's development and its adoption by the user.

One of the latest addition to this area is the DHIRD [5] system. This system is designed for accepting queries in English as well as the Hindi language. This system makes use of the Stanford Parser [6] and the Hindi Shallow Parser [7] for English and Hindi language, respectively. The developed system uses its own Tokenizer so as to run Hindi Shallow parser on Windows platform whereas earlier the parser was only compatible to work upon the Linux environment.

3 Architecture

The architecture of this Web Based Hindi Language Interface to Database using Machine Learning Approach mainly consists of three basic components namely Linguistic, Query Translator & the Query Executor Module. These components are illustrated in the Fig. 1.

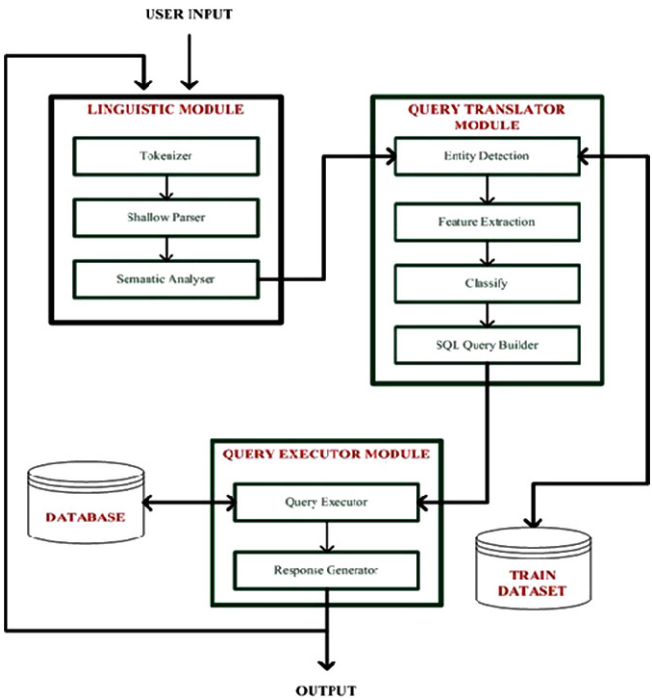

Fig. 1. Architectural representation

3.1 Linguistic Module

All the tasks related to the language of the query such as query entering, preprocessing of the query, semantic analyzing are dealt in the linguistic module. This linguistic module further has three components namely Tokenizer, Shallow Parser and Semantic Analyzer. The work of Tokenizer is to break the query in to different useful tokens. Shallow Parser is used for parsing the input query and for generating the syntax tree. The output of the shallow parser acts as input to the Semantic Analyzer. All the useless tokens are discarded. Matching of the useful tokens with their corresponding English word is done by the Semantic Analyzer. Therefore, important information such as table name, field name (columns name), conditions, function etc. are distinguished by the Semantic Analyzer.

3.2 Query Translator Module

The Query Translator Module has a wide role to play in this system. Query translator module has four components namely Entity Detection, Feature Extraction, Classify, SQL Query Builder. Entity Detection is responsible for the implementation of the similarity check, so as to closely match the misspelled or incomplete words with the correct words.

Feature Extraction is responsible for the generation of feature matrix based on the number of known tokens in each query. On the basis of the output from the Feature Extraction component, classification is done using the Matrix along with the concept of K-Nearest Neighbors (KNN) Algorithm [8].

SQL Query Builder is responsible for creating the queries which are complete in SQL aspect with the help from the previous components. For example:

Input Query: उन सभी वधिार्थीयो के अनुकर्मान्क और अंक बताओ जनिका नाम दीपक है

SQL Query: select rollno, marks from students where name = ‘deepak’

3.3 Query Executor Module

The Query Executor Module comprises mainly of two subcomponents Query Executor and Response Generator. The aim of the Query Executor is to take the SQL query from the Query Builder, the part of Query Translator Module and establishes a connection so as to execute the query on the given database which is provided to the Query Executor. Depending on how the system is being used, Response Generator has various functionalities such as displaying the result in the correct format, finding the accuracy rate or assisting in the training of the system. The Result will be shown as output to the user. In case the system is being used for training purpose then the result from the Response Generator will be again fed to the system itself as described in Fig. 1.

4 Implementation

4.1 Web-Based Interface of the System

Using web techniques such as JSP, Eclipse IDE, Apache Tomcat Server a special Graphical User Interface has been developed that is Web Based so as to increase the scope as well as making it more user friendly. Using the Web Based Graphical User Interface end user enters the input sentence in Hindi language and gets the results in the Hindi language as well. The Hindi language interface shows a number of fields like the Successful Queries Dropdown, Query Textbox, Keyboard Button, Search Button and a Result area. This developed interface is shown in the Fig. 2.

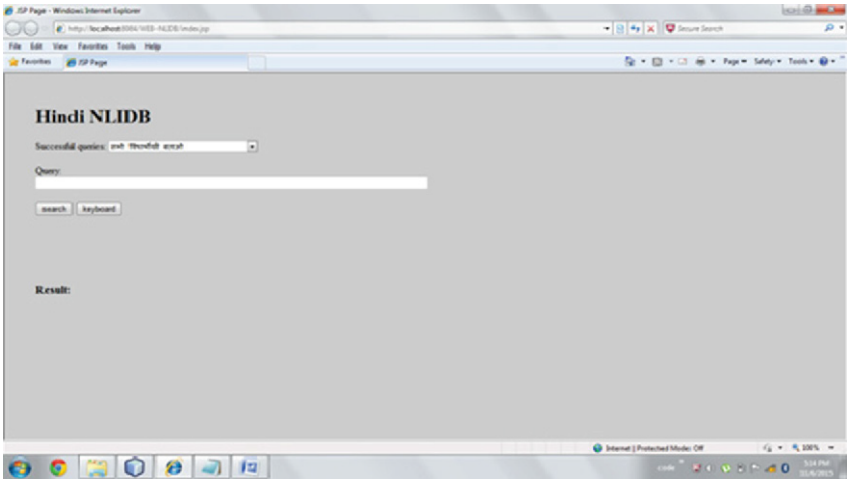

Fig. 2. Graphical user interface

4.2 Keyboard and Search Button

Keyboard has also been included in the developed system so as to make typing of the query much easier and simple. On clicking of the Keyboard Button a keyboard of the helping words or tokens open up. Using this user can simply click on the predefined given words instead of typing the words, the clicked words will directly appear in the Query Textbox. Search Button has been provided to start the query execution process. The query gets executed on clicking the Search Button and the result get displayed as the output. The functionality of the Keyboard as well as the Search button has been illustrated below in Fig. 3.

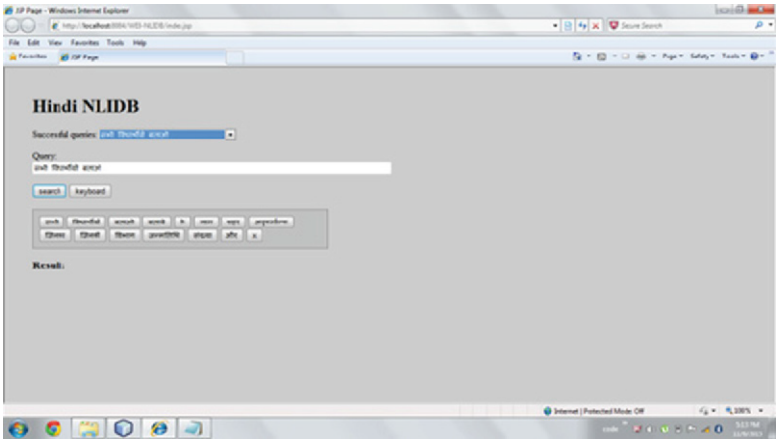

Fig. 3. Keyboard and search button

4.3 Auto Complete Query Textbox

Auto Complete Query Textbox is in place for editing the query or for writing a new query. An additional feature of Auto Complete has been introduced along with the Query Textbox with the help of Jquery. This is shown in the Fig. 4.

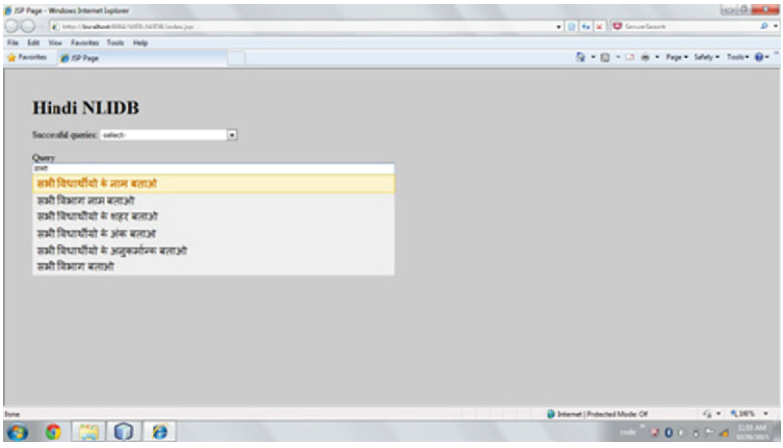

Fig. 4. Query textbox with AutoComplete feature

4.4 Successful Queries Drop Down

Successful Queries Drop Down is based on the query execution history and contains list of all the queries that have been executed effectively. Figure 5 shows the Successful Queries Drop Down list.

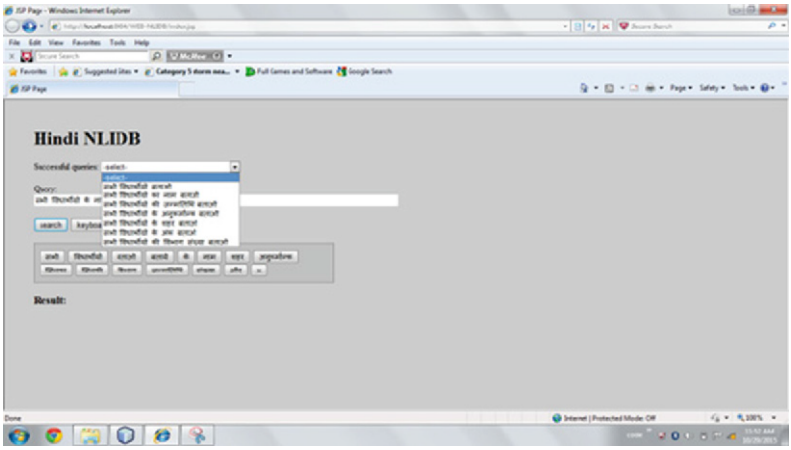

Fig. 5. Successful queries drop down list

5 Testing and Results

For two strings, it describes the similarity of two strings to the length of the longest prefix common to both strings. When the user types the query in the question field sometimes only half the word gets typed or the word is misspelled. Then, from the String Similarity measure working at the back-end, the String Matching function gets the actual word and retrieves the answer according to the question.

The Similarity Functions used to check the accuracy of the system are Euclidean Similarity [9], Jaccard Similarity [10], Dice Similarity [11], Cosine Similarity [12], Smith Waterman Similarity [13], Levenshtein Similarity [14] functions. Results according to these Similarity Functions for multi phrase word and misspelled words matching are based on experiment.

The word used to evaluate these different Similarity functions along with the accuracy rates on different Similarity functions shown in Table 1 is वधिार्थी नाम..

Table 1. Accuracy values of different similarity functions

	विधार्थी नाम	विधार्थी-नाम	विधर्थी नम	विधार्थी	नाम
	T1	T2	T3	T4	T5
Cos. Sim.	1	0	0	0.70710677	0.70710677
Dice Sim.	1	0	0	0.6666667	0.6666667
Euc. Dist.	1	0.22540335	0.2928932	0.5527864	0.5527864
Jacc. Sim.	1	0	0	0.5	0.5
S.W Sim.	1	0.8333333	0.9	1	1
Lev. Sim	1	0.9166667	0.8333333	0.6666666	0.25

Graph associated with these implemented accuracy values is shown in Fig. 6 below.

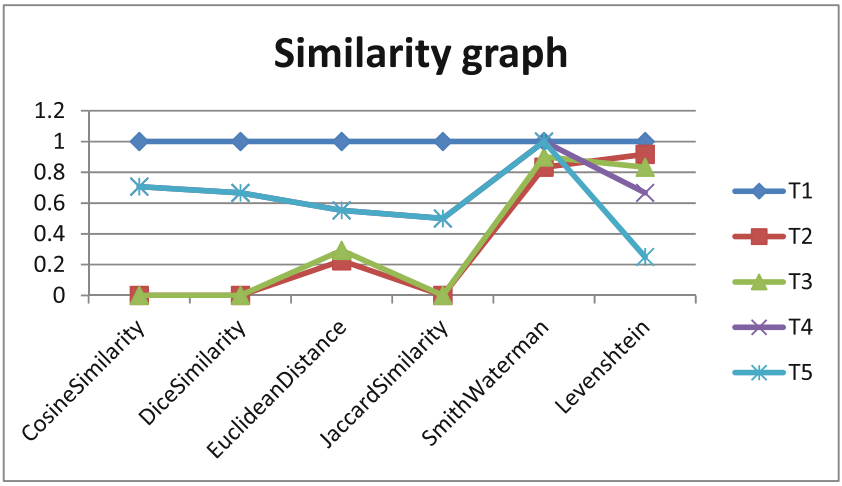

Fig. 6. Similarity function accuracy graph

Depending on the above shown results as well as the graph, Smith Waterman Similarity function is selected as the most suitable Similarity function as it has the most accurate results for most of the test cases. The system described in the paper has been implemented using the Smith Waterman Similarity function.

The result of the query execution is illustrated below in Fig. 7.

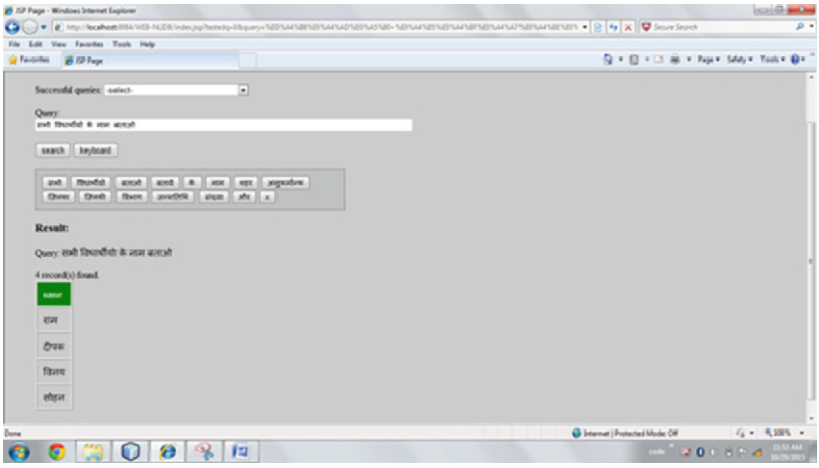

Fig. 7. Snapshot of executed query result

6 Conclusion and Future Work

The paper discusses the implementation of Advanced Web-Based Hindi Language Interface to Database using Machine Learning approach. The Similarity functions have been tested on the developed system to make the system more accurate. The system has been made Web Based and has been provided with the functionality of Auto Complete feature to increase the scope and make it more user friendly respectively. The future scope of the implemented system can have use of multiple languages in a single system along with inclusion of data base independent characteristics.

References

1. Woods, W.A., Kaplan, R.M., Webber, B.N.: The lunar sciences natural language information system: Final Report. In: BBN Report 2378, Bolt Beranek and Newman Inc., Cambridge, Massachusetts (1972)
2. Shingala, P.V.: Enhancing the relevance of information retrieval by querying the database in natural form. In: International Conference on Intelligent Systems and Signal Processing (ISSP), pp. 408–412. IEEE (2013)
3. Khalid, M.A., Jijkoun, V., de Rijke, M.: Machine learning for question answering from tabular data. In: 18th International Workshop on Database and Expert Systems Applications, pp. 392–396, DEXA (2007)
4. Dua, M., Kumar, S., Virk, Z.S.: Hindi language graphical user interface to database management system. In: 12th International Conference on Machine Learning and Applications, pp. 549–554. IEEE, Florida (2013)
5. Kumar, R., Dua, M., Jindal, S.: Domain-independent Hindi language interface to relational database. In: International Conference on Computation of Power, Energy, Information and Communication, pp. 81–86. IEEE (2014)
6. Stanford Parser. http://nlp.stanford.edu/software/index.shtml
7. Hindi Shallow Parser. http://ltrc.iiit.ac.in/analyzer/hindi/
8. Guo, G., Wang, H., Bell, D.J., Bi, Y., Greer, K.: KNN model-based approach in classification. In: Meersman, R., Schmidt, D.C. (eds.) CoopIS/DOA/ODBASE 2003. LNCS, vol. 2888, pp. 986–996. Springer, Heidelberg (2003)
9. Marukatat, S., Methasate, I.: Fast nearest neighbor retrieval using randomized binary codes and approximate euclidean distance. J. Pattern Recognit. Lett. **34**, 1101–1107 (2013). Elsevier
10. Sarawagi, S., Kirpal, A.: Efficient set joins on Similarity Predicates. In: International Conference on Management of Data (SIGMOD 2004), pp. 743–754. ACM (2004)
11. Ye, J.: Multicriteria Decision-Making method using the Dice Similarity Measure based on the Reduct Intuitionistic Fuzzy sets of Interval-Valued Intuitionistic Fuzzy Sets. J. Appl. Math. Modell. **36**, 4466–4472 (2012). Elsevier
12. Sahami, M., Heilman, T.: A web-based kernel function for measuring the similarity of short text snippets. In: 15th International Conference on World Wide Web, pp. 377–386. ACM (2006)

13. Mott, R.: Maximum likelihood estimation of the statistical distribution of smith waterman local sequence similarity scores. Bull. Math. Biol. **54**, 59–75 (1992). Springer
14. Cohen, W., Ravikumar, P., Fienberg, S.: A comparison of string metrics for matching names and records. In: American Association for Artificial Intelligence (2003)