Natural Language SQL Query Processing using Fuzzy Matching and Elimination Technique

Praveena Mydolalu Veerappa, Ajeet Annarao Chikkamannur

Abstract*: *In Structured Query Language (SQL), complex queries are difficult to write or understand by a user, because every user is not familiar with SQL. A common user can able to retrieve the information from the query databases using natural language is considered as an important research area. To improve the communication between databases application and naive user, an enhanced application with intelligent interface are needed. A fuzzy system with matching and elimination technique is designed in this research study, where SQL queries are formed from the input given by the user through several steps like noise removal, lexicon normalization and query formation. Then, the system uses the Latent Dirichlet Allocation (LDA) to extract the keywords from the input query. Finally, matching and elimination techniques are used to find the data, which is related to the input query given by end-user. When compared with the existing SQL techniques, the proposed fuzzy method achieved 91% and 90.5% accuracy, 95% and 93% precision, and 0.10 and 0.12 error rate for both 28 and 50 queries.

Keywords*: *Elimination Technique, Fuzzy Matching Technique, Natural Language, Query Database, Structured Query Language.

I. INTRODUCTION

Nowadays, the information is highly available in World Wide Web (WWW), however the retrieval of searched information does not match with the user's query by using traditional search methods. At present, people are highly interested to extract the precise information from the web. Therefore, during the extraction of precise information from WWW, an effective and efficient mechanisms are required [1,2]. The natural language must be understanding by the system and able to interact with the data ware house to retrieve the information accordingly. The automatic analysis and representation of human language can be described as theory-motivated range of computational techniques called Natural Language Processing (NLP) [3]. The extraction of data from the data ware house by using various solutions of user interface, which can be categorized as Natural language based query interface, form-based interface and keyword-based interface. The interactive user interface is provided by form-based interface, but if provider failed to provide the proper details for extracting the data (i.e. possibly manipulated by SQL), this system achieved poor performance than other interfaces [4]. The user can access the data from the databases by entering the query in either English or any other language, which can be allowed by Keyword or Natural Language based interface system. However, only experts uses the SQL query commands for retrieving the data from the storage application. The other common people who wants to retrieve the data must require a system, which can access the natural language and provides the proper information [5]. The process of converting the input query into SQL query, researchers developed the methods for applying semantic, lexicon and syntactic analysis to make the better interaction between database and non-professional [6,7].

In several fields namely pattern recognition and computer vision, Machine Learning Algorithms (MLA) made an effective advance. At present, the use of MLA gained a much interest among the community for NLP research. More than several decades, the NLP problems are targeted by MLA according to shallow models i.e. logistic regression and Naive Bayes and trained on sparse and high dimensional features [8]. According to dense vector representation, the neural networks provided better performance on various NLP tasks in the last few years. This is due to the success collaboration of deep learning methods and word embedding. The multi-level automatic features representations are enabled by deep learning techniques. However, traditional MLA based NLP systems are highly dependent on hand-crafted features. These features are always incomplete and time-consuming, which is the major limitations of existing MLA on NLP tasks [9,10]. A fuzzy query is made and pre-processed by providing a user interface in this research work. The linguistic terms are expressed along with SQL basic language by using these fuzzy queries. The imprecise terms namely “medium”, “approximately-equal-to”, etc. are identified by pre-processing these fuzzy queries and promote the threshold values to define the fuzzy operators. The fuzzy relationships are satisfied by retrieving all the records using the pre-processed module. The basic capability of any relational query language are formed by this research study, where the users formulate the queries in precise terms to retrieve the information from the databases.

The remaining portions of paper consists of: Section 2 discuss the review of existing techniques with its advantages and limitations. Section 3 defines the explanation of SQL, where the brief description of proposed method is given in Section 4. The experiments are conducted to validate the effective of proposed fuzzy method against existing techniques is presented in Section 5.

Revised Manuscript Received on December 12, 2019.

***** Correspondence Author

Praveena Mydolalu Veerappa*, Department of Computer Science and Engineering, Dr Ambedkar Institute of Technology, Bangalore, India. Email: mv_praveena@rediffmail.com

Dr. Ajeet Annarao Chikkamannur, Department of Computer Science and Engineering, R L Jalappa Institute of Technology, Bangalore, India. Email: ac.ajeet@gmail.com

Finally, the conclusion of the research work with future development is depicted in Section 6.

II. LITERATURE REVIEW

In this sub-section, a review of existing techniques is discussed, which are developed to transform the natural language into structured query language. The advantage of the existing techniques with its limitations are also discussed in [11-16].

A. Solanki, and A. Kumar, [11] designed a three-tier architecture to handle the Natural Language Query (NLQ) processing and transferred those NQL into SQL by using semantic and pattern matching. The three-tier system consists of user interface, database and processing unit for better understanding the SQL to end users. The input was processed in several steps namely, escape words removal, tokenization, elements classification and finally, query formation. The final output of the three-tier architecture was in the form of a SQL query. The experiments were carried out to validate the performance of the method by means of accuracy, precision and recall against existing system NLT-SQLC [12]. However, when there was an increase in NLQs, the performance of precision showed slight decrease in the precision value.

M. S. Ramada, J. C. da Silva, and P. de Sa Leitao-Junior, [13] enabled the specification of queries by implementing the system called Semantic Keyword Query System for Information Retrieval from Relational Databases (SQUIRREL). The important aspects namely use of aggregate functions, proximity between keywords and query segmentation were considered by SQUIRREL. When compared with existing techniques, this approach produced a smaller and significant set of results for the keyword queries. The usage of single inverted commas identified the composite values of SQUIRREL method, which required the user's priori knowledge for constructing the query. However, the method didn't analysis how the queries were obtained to validate their effectiveness.

X. Hu, D. Dang, Y. Yao, and L. Ye, [14] developed a Natural Language Aggregate Query (NLAQ) to automatically identify the aggregation and transformed it to SPARQL aggregate statement. In addition, an extended paraphrase dictionary ED was designed to build a bridge between RDF data and query intention and also obtained more candidate mappings for semantic relations. The inappropriate candidate mapping combinations were filtered out in basic graph patterns and semantic relations by introducing a predicate-type adjacent set PT. The experiments were carried out on two real datasets namely DBpedia and QALD benchmark for validating the effectiveness of NLAQ. But, the method didn't concentrate on how to answer an implicit query and also difficult to identify the semantic relations between all NLQ.

R. S. Yadav, [15] handle the SQL processing problems by presenting the fuzzy and vague sets, where the crisp sets were converted into vague set by developing the Positive Ordered Transformation formula (POTF). The fuzzy information was obtained by converting the vague information using the method Transforming Vague Set into Fuzzy Set method. In addition, according to SQL query processing, similar tuples for vague, classical fuzzy and converted fuzzy sets were obtained by calculating the similarity measures. When compared with fuzzy sets based method to handle the SQL query, this approach delivered more exact results during the experimental validations. But, the fuzzy logic only deals with partial information and didn't concentrate on exact values.

P. S. Banerjee, B. Chakraborty, D. Tripathi, H. Gupta, and S. S. Kumar, [16] designed a Natural Language Information Interpretation and Representation System (NLIIRS) to process the information and allowed user to retrieve it by asking any “Who”, “Where” and “When” type of questions. The use of SQL at the back-end for information storage, extraction and processing were avoided by NLIIRS. After every execution step, the extraction process and pattern matching of the answers to the queries became faster and concise. The major advantage of using NLIIRS system was that it didn't require training and also enable the user for retrieving any information from the available unstructured information. There were some limitations exists in the NLIIRS method: The verb of the sentences is not correctly identified by this system. The compound sentences include "and" and other conjunctions are not properly explained by this system. Moreover, the data format only in numeric form is processed by this NLIIRS.

III. DEFINITION OF SQL

The four main statements to change the data from the ware houses are provided by Oracle, which is classified into INSERT, MERGE, DELETE and UPDATE. The advanced options are presents in each of these statements to specific the commands. Some additional special commands include ROLLBACK, TRUNCATE, COMMIT, ALTER SESSION, SAVEPOINT and so on to change the data. The changes and modifications made in the data are not stored in the Oracle, which is a challenging problem, and it contains several options for file input and output. The three major clauses present in the any SQL query such as SELECT, WHERE and FROM. The retrieved columns are isolated from the database by using the SELECT clause. To specify and retrieve the data/tables from sample database, a FROM clause is used [17,18]. The retrieved rows must satisfy the conditions with a Boolean predicate by using a WHERE clause. The disjunctive normal form is used to transform all the WHERE predicates, where this form is also called as a disjunction of conjunctions of clauses. A conjunctive predicate is used to connect with OR operators and also connects the clauses with AND operators, which is defined as smaller predicate. The column opr column or column opr constant are formed by each clause, where comparator is defined by opr. In this research study, some special keywords are also studied, which are stated as follows:

The right types of tables are created with right properties, then the researchers should know how to modify those tables by using proper alter properties and drop the tables. The syntax for ALTER TABLE is non-transactional and massive, which requires a high downtime.

The user can change the tables without significant downtime using online table redefinition. However, dropping the table is easy and simple, it is obviously dangerous. The table can be put in the recycle bin by using the syntax includes DROP TABLE_NAME or REMOVE TABLE_NAME. The attributes such as archives or purge are also used to remove the statement from the table. The space obtained in the recycle bin can be able to enable by default, however research always should check the parameters of RECYCLEBIN [19,20]. Still, the table uses the same amount of disk space for process, until and unless the table is purged from the recycle bin. Whenever the keyword PURGE is added to the end of the command, the space should reclaim immediately and also skip the recycle bin. Permanently, the data objects are dropped by using the PURGE options or simply adding the following commands to the end of the statements such as PURGE DBA_RECYCLEBIN or PURGE USER_RECYCLEBIN. The other commands such as undo, redo and temporary table space are considered as the non-permanent segment types in the SQL query.

IV. PROPOSED METHODOLOGY

NLP is the process of analyzing, extracting and understanding the human language meaning in an effective way, which are used for computers to interact with the database. This interaction is used to retrieve the information which are more likely to input query given by the end-user. The ordinary user faces the difficulty to understand and write the SQL query due to continuous increase of database sizes and complexity in the relation among the entities. The details of database include entities, relations and so on are also known by the user. But, the major problem of the user who want to retrieve the data from the ware house, are unable to form a SQL queries. To solve the issues, this research study develops the fuzzy logic based operations, which provides an interface to the end user. Fig. 1 shows the working procedure of proposed method.

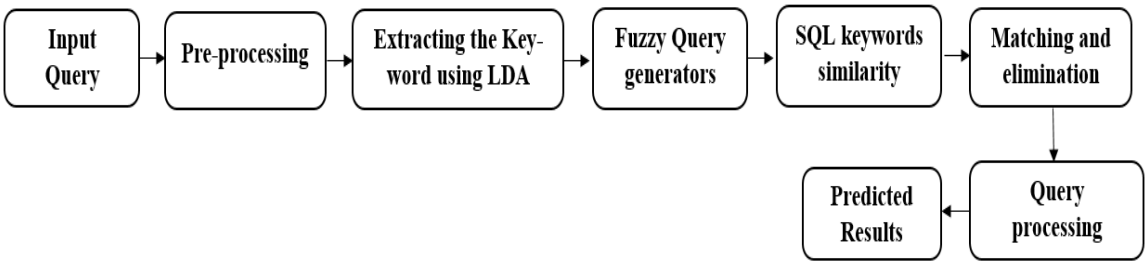

Fig. 1.Working Procedure of Proposed Method

Table 1 shows sample database used for this research study, where the query is given in the following sub-sections.

Table- I: Sample Database (F: Female, M: Male)

S. No	Name	Age (yrs.)	Gender	Height (ft)	Weight (kg)
1	Gita	23	F	4.7	30
2	Ankit	25	M	5.2	30
3	George	35	M	5.7	53
4	Rita	23	F	4.7	24
5	Sonu	27	F	5.3	45
6	John	40	M	4.9	38

A. Pre-Processing

Before applying any MLA, the preparation of the data is the most important stage. The preparation of query is the preliminary step for processing the input query and then, noise should be clean by removing the useless words [21]. The pre-processing steps are conducted in this study by using several steps includes noise removal, parsing of entity and lexicon normalization, which are discussed as below:

i) Noise Removal: Noise is defined as the fragments of text, which are not relevant to the final results in the text query. All different types of noises are removed in this part of research, which are available in the input query.

ii) Lexicon normalization: The multiple various representations are used to demonstrate the words is called as noise of the text query. The words' variants and all forms are transformed into normalized and uniform forms by using the stage of lexical normalization. The original dictionary form (i.e. lemma) are obtained from the word by the transformation process called lemmatization, which is close to stemming. The semantic canonical form of the words is brought by the lemmatization technique, i.e. nominative case of the singular for adjectives and nouns, infinitive for the verb. A stemmer is used to quickly find the words basis in the given text query.

iii) Parsing of entity: The entities are defined as the verbs and nouns phrases, which is the most important parts of the sentences. While analyzing the dependent words, the entities in the text are identified according to rules. The Part-Of-Speech Tagging is usually defined as automatic part-time text markup in this process. In the phrase and sentence, the context of words includes nearest neighboring words, general context via subject matter of the text and distant grammatically related fragments are considered by the analyzer. After the pre-processing step, the keywords are extracted by using LDA, which are explained in the following sub-section.

B. Keyword Extraction using LDA

The LDA approach is used to extract the keyword once the pre-processing of collected data is finished [22,23]. A probabilistic topic model is also called as LDA, where random mixture of latent topics is also defined as every query in the text. In LDA, the labels of every latent topic are distributed over a fixed set of words, which is used to estimate the underlying latent topic structure on the basis of observed data.

Normally, for every document, the words generations are conducted in the two-phase process. A distribution over topics is randomly selected in the first phase for those documents.

In LDA, a word is a distinct data from a vocabulary index $\{1,...V\}$, a series of N words are W.

During the corpus generation, the observation of LDA is carried out for the three layered representation, then two parameters such as π *and* μ are examined. The investigation of document-level topic variables is conducted for each and every document. In LDA, the examination of word level variables is conducted for each word of document. The generative process of LDA presents the joint distribution over random variable. Eq. (1) is used to find the probability density function of the $k-$dim*ensional* dirichlet random variable. Successively, the joint distribution of a topic mixture and the probability of a corpus are evaluated by using the Eq. (2) and (3).

$$p(\aleph \mid \pi) = \frac{\Gamma\left(\sum_{i=1}^{k} \pi_i\right)}{\prod_{i=1}^{k} \Gamma(\pi_i)} \aleph_1^{\pi_1 - 1} ..... \aleph_k^{\pi_k - 1} \tag{1}$$

$$p(\aleph, x, y \mid \pi, \mu) = p(\aleph \mid \pi) \prod_{n=1}^{N} p(x_n \mid \aleph)\, p(y_n \mid x_n, \beta) \tag{2}$$

$$p(D \mid \pi, \mu) = \prod_{d=1}^{M} \int p(\aleph_d \mid \pi) \times \left(\prod_{n=1}^{N_d} \sum_{x_{dn}} p(x_{dn} \mid \aleph_d)\, p(y_{dn} \mid x_{dn}, \mu) \right) d\aleph_d \tag{3}$$

Where, document-level topic variable is illustrated by $\aleph$ dirichlet parameter is described by π, document is represented by M number of words is presented as N, the topics are characterized by μ, per-word topic assignment is denoted by x and observed word is indicated by y.

The LDA model's important inferential task is to calculate the posterior distribution of the hidden variable in a given query. The intractable problem is considered as the posterior distribution of exact inference of the hidden variable. While combining the LDA with suitable techniques namely Gibbs sampling, Markov chain, Laplace, and variational approximation are extensively utilized for keyword extraction. An individual weight value is used to extract the text query keyword and stored in dictionary. In testing phase, the dictionary is coordinated with the testing text query to attain the weight value.

C. Generations of Fuzzy Query

According to query definitions, the fuzzy queries can be categorized into three types, where such different queries are explained with samples are given as below:

i) Simple Query: Consider the sample as "select * from sample where age is small", which resembles the simple query. For instance, the impreciseness meaning are shown by the word "small" from the above query. Therefore, the projections on attributes are considered by this simple queries.

ii) Complex Query: Let consider other sample as "select * from sample where weight is around 50 and height is medium", where the two terms 'around' and 'medium' are defined as complex query.

iii) Quantified Query: Consider the sample: "select * from sample where most of the important attributes out of {age is very large, weight is approx_equal to 60, height is around 5} match". In this example, the terms 'most', 'important', 'very large', 'approx_equal' and 'around' are also imprecise in nature. In addition, context sensitive is presented in these imprecise terms. For instance, if the attribute "age" is changed into "height" of the student as shown in the simple query sample, the usage of the term "small" meaning will be changed.

The condition of a "selection" operation or formula of a relational algebra with the difference are viewed by fuzzy query. The real interval [0,1] defines the formula for falsehood degree or partially truthness, where true represents the formula by 1 and false by 0. The fuzzy predicates and logical operators (i.e. and, or, negation) are presents in the logical infix expression, which are considered by this formula. The fuzzy selection operators are used to form these fuzzy predicates. Therefore, the problem of validating this logical infix expression is viewed as same as the problem of interpreting a fuzzy query. A fuzzy select pre-processor is used for validating this expression.

The input query is accepted from the end-user and verify it for syntax errors by using query analyzer module. The "fuzzy query pre-processor" as FQP obtained the query, when there is no syntax error is present in the input data. The user obtained the error message from the modules, suppose there is any syntax errors presents. The user queries are read out from the input file by FQP, which is the kernel of the proposed preprocessor. The tokens or basic elements are separated and the lexical phase is used to scan the query. At last, the syntactic classes are recognized by the terminal table.

D. SQL keywords Similarity

Every terminal symbol and their classification has an entry in a terminal table, which is a permanent database. Terminal symbols include reserved words, fuzzy selection operators, special characters etc. [24,25]. The logical infix expression is used to define the semantics of all fuzzy selection operators, once the fuzzy operators make a call to the similarity function of SQL. The postfix notations are converted from the infix expression, where these expressions are parsed by operators. Later, "the matching and elimination unit" are used to handle the postfix notation. For instance, in a query "age is small", the imprecise term "small" is also shown by the attribute name "age". According to the subjective perception of the user group, the elicitation operations completely works, where the minimum similarity degree are elicited to allow in the fuzzy selection formula. This helps the records to have the higher similarity degree or equal to this elicited value in the output files. The real numbers in [0,1] is considered as the matching degree and also shows the similarity degree, where false is represented by "0" and true is illustrated by "1".

E. Matching and Elimination

In this unit, step through the records of a given type, one at a time. This unit obtained the query in postfix notation i.e. by a FQP, when a new record is accessed every time. Over this postfix expression, a scan is made by the unit at the time of evaluation, then the fuzzy selection operator is evaluated from the accessed records, when the operands stacks values are identified. Finally, the record's final truthness degree is estimated, here the similarity degree of a record is also defined as truthness degree. When compared with the pre-specified threshold value, the similarity degree for one record is identified, where these threshold value is elicited by the elicitation unit. The record is fed into an output file, once their similarity degree is greater than the pre-specified threshold value. Suppose, if it is less than the threshold value, then the record is rejected. At last, the records which closely match the query are presented in the output files. From the evaluation of the fuzzy query, the results are stored in this output files. Therefore, the records in the output files must have the similarity degree, which is greater than or equal to the threshold value, where these values are neglected by the elicitation unit.

By using the proposed method, the following query can be solved as:

The query 1: SELECT name FROM Sample WHERE height < 4.9 and weight > 25.

The proposed fuzzy method processes this query and obtain the results as shown in Table 2.

Table- II: Retrieved records from the sample database

S. No.	Name	Age (yrs.)	Gender	Height (ft)	Weight (kg)
1	Gita	23	F	4.7	30

The query 2: REMOVE the data who's having the age 27 FROM Sample.

The proposed fuzzy method processes this query and the keyword REMOVE can be replaced by PURGE keyword. If any end-user gives PURGE keyword instead of REMOVE, then also the proposed method processes the query and identified the results, which is illustrated in Table 3.

Table- III: Results for the query 2

S. No	Name	Age (yrs.)	Gender	Height (ft)	Weight (kg)
1	Gita	23	F	4.7	30
2	Ankit	25	M	5.2	30
3	George	35	M	5.7	53
4	Rita	23	F	4.7	24
5	John	40	M	4.9	38

The above table presents the results for the query given by the end user. But, the name SONU data is deleted from the sample data by using these query.

V. RESULTS AND DISCUSSION

In this section, the proposed fuzzy method is validated against existing techniques by means of recall, precision, accuracy and error rate. In this experimental setup, the proposed fuzzy method was simulated by using Python 3.7.3 with 8GB RAM, Intel Core i5 with 2.2 GHz. The proposed method uses the sample database records as a collected data to validate its effectiveness against various other techniques. The following below section explains the parameter evaluation, quantitative and qualitative analysis of proposed fuzzy method.

A. Parameter Evaluation

The proposed fuzzy method is validated against existing techniques by using several parameters, which are discussed in this section. The property of the proposed fuzzy system is verified and also check the practical and theoretical developments of those systems by using the evaluation metrics. It follows the common underlying methodology for validation, which consists of a set of various measures. Some of the major parameter metrics are used to evaluate the purpose are recall, precision, error rate and accuracy. The equations from Eq. (4-7) are used for calculating these metrics, which are given below,

$$\Pr ecision = \frac{TP}{TP + FP} \tag{4}$$

$$\text{Re}call = \frac{TP}{TP + FN} \tag{5}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \tag{6}$$

$$Error\ rate = \frac{FP + FN}{Total\ number\ of\ queries\ fired} \tag{7}$$

Where, TP is denoted as true positive, TN is indicated as true negative, FP is represented as false positive, and FN is denoted as false negative. In the below section, the proposed method is validated against existing techniques.

B. Performance Analysis of Proposed Fuzzy Method

The performance of the proposed fuzzy method is validated in terms of accuracy, precision, recall and error rate against NQL-SQL [11], NLT-SQLC [12] and NLAQ [14] for 28 and 50 queries. The existing NLAQ are implemented for this queries to validate the effectiveness of fuzzy system. In Table 4, initially the proposed method is validated for 28 queries by means of all the parameters. Figure 2 and 3 shows the graphical representation of all parameters metrics.

Table- IV: Performance of Proposed Method for 28 Queries

Methodology	Accuracy	Precision	Recall	Error Rate
Existing NLT-SQLC [12]	80	84	80	0.17
NQL-SQL [11]	85	90	83	0.14
NLAQ [14]	86	91.5	84.6	0.15
Proposed Fuzzy Method	91	95	90	0.10

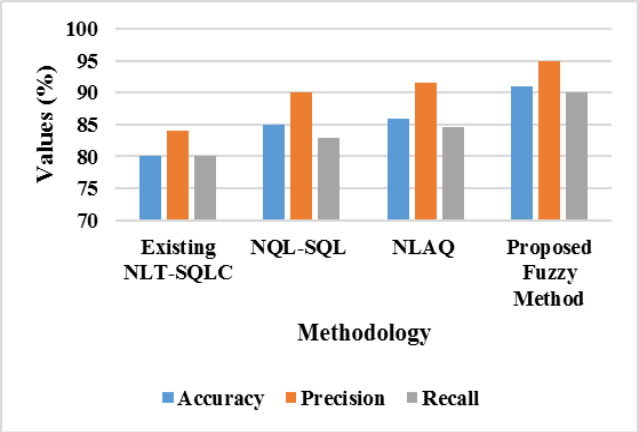

Fig. 2.Performance of Proposed Method by means of accuracy, precision and recall

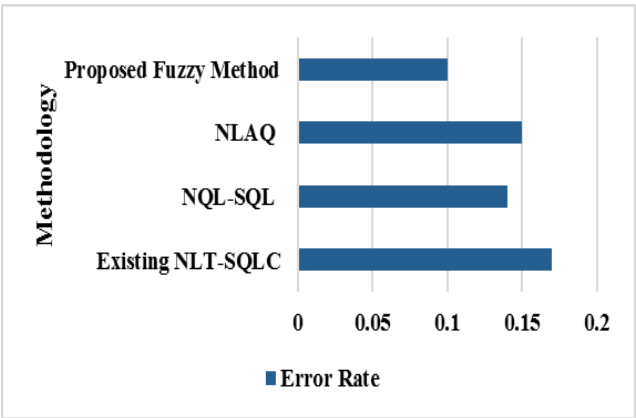

Fig. 3.Performance of Error Rate using Proposed Fuzzy method

From the above table 4 and figures, it is clearly stated that the proposed method achieved better performance than existing techniques: NLTSQLC and NQL-SQL. For 28 queries, the proposed fuzzy method achieved 91% accuracy, 95% precision and 90% recall, where the existing NQL-SQL technique achieved 85% accuracy, 90% precision and 83% recall. However, the error rate of both techniques are high for 28 queries. For instance, the proposed fuzzy method gained 0.10% error rate, while comparing with existing techniques. The proposed method achieved higher performance due to FQP. The matching and elimination of the query are processed before storing the data. The same experiments are conducted against existing techniques by using all parameter metrics for 50 queries. The validated results are presented in Table 5 and the graphical representation are given in Figure 4 and 5.

Table- V: Performance of Proposed Method for 50 Queries

Methodology	Accuracy	Precision	Recall	Error Rate
Existing NLT-SQLC [12]	80	86	80	0.20
NQL-SQL [11]	84	89	80	0.16
NLAQ [14]	86	91	82	0.15
Proposed Fuzzy Method	90.5	93	89	0.12

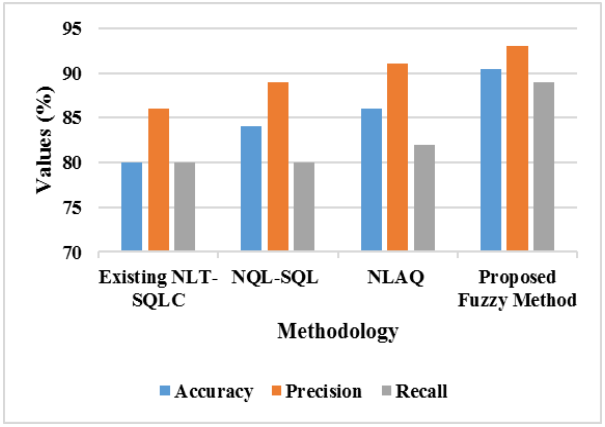

Fig. 4.Performance of Proposed Fuzzy Method against existing techniques

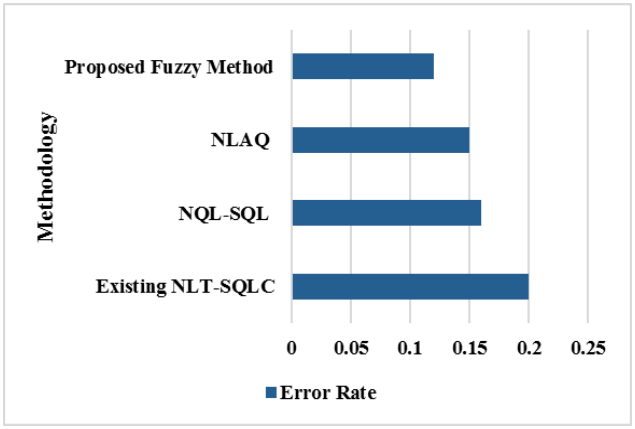

Fig. 5.Error Rate Analysis of Proposed Fuzzy Method

The experiments for 50 queries showed that the proposed fuzzy method achieved higher performance than other two existing techniques. The NLT-SQLC and NQL-SQL techniques achieved nearly 85% accuracy, 88% precision and 80% recall, but the proposed fuzzy method achieved 90.5% accuracy, 89% recall and 93% precision for 50 queries. While experimenting on error rate, the proposed fuzzy method achieved 0.12, where the existing techniques obtained 0.20 and 0.16 error rate for 50 queries. However, the error rate of proposed fuzzy method is high for 50 queries, when compared with 28 queries. The overall performance of proposed fuzzy method for both 28 and 50 queries are validated and presented in Table 6.

Table- VI: Overall Performance of Proposed Fuzzy Method

No. of Queries	Accuracy	Precision	Recall	Error Rate
28	91	95	90	0.10
50	90.5	93	89	0.12

From the Table 6, it is clearly stated the performance of proposed fuzzy method for both 28 and 50 queries. However, there is a slight performance changes of proposed fuzzy method for all parameter metrics, when the number of queries increases. For instance, error rate is 0.12 for 50 queries and

0.10 for 28 queries, where proposed fuzzy method achieved 90% recall for 28 queries and 89% recall for 50 queries.

VI. CONCLUSION

To handle the huge number of data, database is an effective solution, where user must learn the SQL to understand the structure of these databases. The users who are not familiar with the SQL to interact with the database for information retrieval, need a system for better interaction. According to the natural language query given by the user, the system must able to interact with the database effectively. An efficient fuzzy system is proposed in this research study to handle the challenges occurred in NQL processing. The evaluation of correct SQL translations is the main aim of this study for the given input user's NLQ. The keyword from the query are extracted by using LDA method and these keywords are given as input to fuzzy matching technique. The proposed pre-processor is linked with the created sample relational database attributes. The SQL queries are used to retrieve the records one by one from these created databases. According to the pre-specified threshold value and matching degree, the fetched records are selected. The experimental results stated that the proposed fuzzy method achieved 91% accuracy, 95% precision, 90% recall and 0.10 error rate for 28 queries and 90.5% accuracy, 0.12% error rate, 89% recall and 93% precision for 50 queries when compared with existing technique NQL-SQL. This work is given for a basic three types of queries in SQL structure, however this work can be extended to further different types of query architectures.

REFERENCES

1. A. Hogan, A. Harth, J. Umbrich, S. Kinsella, A. Polleres, and S. Decker. (2011). Searching and browsing linked data with swse: The semantic web search engine. *Web semantics: science, services and agents on the world wide web*, 9(4), pp. 365-401.
2. A. J. Shaikh, and V. L. Kolhe. (2013). Framework for web content mining using semantic search and natural language queries. *IEEE International Conference on Computational Intelligence and Computing Research.*
3. M. Egea, and C. Dania. (2019). SQL-PL4OCL: an automatic code generator from OCL to SQL procedural language. *Software & Systems Modeling*, 18(1), pp. 769-791.
4. B. Sujatha, and S. Viswanadha Raju. (2016). Ontology based natural language interface for relational databases. *Procedia Computer Science*, 92, pp. 487-492.
5. M. Fragkoulis, D. Spinellis, and P. Louridas. (2015). An interactive SQL relational interface for querying main-memory data structures. *Computing*, 97(12), pp. 1141-1164.
6. N. Nihalani, S. Silakari, and M. Motwani. (2011). Natural language interface for database: a brief review. *International Journal of Computer Science Issues (IJCSI)*, 8(2), pp. 600.
7. A. Shah, J. Pareek, H. Patel, and N. Panchal. (2013). NLKBIDB-Natural language and keyword based interface to database. *In 2013 International conference on advances in computing, communications and informatics (ICACCI)*, pp. 1569-1576.
8. S. Han, H. Shim, B. Kim, S. Park, S. Ryu, and G. G. Lee. (2015). Keyword question answering system with report generation for linked data. *In 2015 International Conference on Big Data and Smart Computing (BIGCOMP)*, pp. 23-26, 2015.
9. C. Baik, H. V. Jagadish, and Y. Li. (2019). Bridging the semantic gap with SQL query logs in natural language interfaces to databases. *IEEE 35th International Conference on Data Engineering (ICDE).*
10. P. Pruski, S. Lohar, W. Goss, A. Rasin, and J. Cleland-Huang. (2015). *TiQi: answering unstructured natural language trace queries. Requirements Engineering*, 20(3), pp. 215-232.
11. A. Solanki, and A. Kumar. (2018). A system to transform natural language queries into SQL queries. *International Journal of Information Technology*, 1(10).
12. G. Singh, and A. Solanki. (2016). An algorithm to transform natural language into SQL queries for relational databases. *Self-organizology*, 3(3), pp. 100-116.
13. M. S. Ramada, J. C. da Silva, and P. de Sa Leitao-Junior. (2019). From keywords to relational database content: A semantic mapping method. *Information Systems*, pp. 101460.
14. X. Hu, D. Dang, Y. Yao, and L. Ye. (2018). Natural language aggregate query over RDF data. *Information Sciences*, 454, pp. 363-381.
15. R. S. Yadav. (2019). A study of SQL query processing using soft computing techniques: a hybrid vague logic approach. *International Journal of Information Technology*, 11(2), pp. 393-405.
16. P. S. Banerjee, B. Chakraborty, D. Tripathi, H. Gupta, and S. S. Kumar. (2019). An Information Retrieval Based on Question and Answering and NER for Unstructured Information Without Using SQL, *Wireless Personal Communications*, pp.1-23.
17. W. Huijie. A Security Framework for Database Auditing System. *IEEE 10th International Symposium on Computational Intelligence and Design (ISCID)*, 1.
18. S. Gadgil, S. Pillai, and S. Poojary. (2013). SQL Injection Attacks and Prevention Techniques. *International Journal on Recent and Innovation Trends in Computing and Communication*, 1(4), pp. 293-296.
19. C. Bobed, and E. Mena. (2016). QueryGen: Semantic interpretation of keyword queries over heterogeneous information systems. *Information Sciences*, 329, pp. 412-433.
20. N. Mahmood, S. M. A. Burney, K. Rizwan, A. Shah, and A. Nadeem. (2017). Building spatio-temporal database model based on ontological approach using relational database environment, *Mehran University Research Journal of Engineering & Technology*, 36(4).
21. M. A. Hamada, K. Sultanbek, B. Alzhanov, and B. Tokbanov. Sentimental text processing tool for Russian language based on machine learning algorithms. *In Proceedings of the 5th International Conference on Engineering and MIS*, pp. 37.
22. J. Boyd-Graber, and P. Resnik. (2010). Holistic sentiment analysis across languages: Multilingual supervised latent Dirichlet allocation. *In proceedings of the 2010 conference on empirical methods in natural language processing, Association for Computational Linguistics*, pp. 45-55.
23. F. Colace, M. De Santo, L. Greco, V. Moscato, and A. Picariello. (2016). Probabilistic approaches for sentiment analysis: Latent dirichlet allocation for ontology building and sentiment extraction. *In Sentiment Analysis and Ontology Engineering*, Springer, pp. 75-91.
24. R. Kumar, and M. Dua. (2014). Translating controlled natural language query into SQL query using pattern matching technique. *International Conference for Convergence for Technology*, pp. 1-5.
25. P. Buche, C. Dervin, O. Haemmerlé, and R. Thomopoulos. (2005). Fuzzy querying of incomplete, imprecise, and heterogeneously structured data in the relational model using ontologies and rules. *IEEE Transactions on fuzzy systems*, 13(3), pp. 373-383.

AUTHORS PROFILE

Praveena M V received his Master of Technology degree from Visvesvaraya Technological University, Belagavi, India in 2013. Presently working as Assistant Professor, Department of Computer Science and Engineering, Dr Ambedkar Institute of Technology, Bangalore and teaching for graduate courses. His research interests are Relational Database Management Systems, Software Engineering, Cloud Computing, and Object Oriented Analysis and Design.

Dr. Ajeet A. Chikkamannur received his Ph.D. degree from Visvesvaraya Technological University, Belagavi, India in 2013. Presently working as a Professor, Department of Computer Science and Engineering, R L Jalappa Institute of Technology, Daddaballapur, Bangalore Rural. His research interests are Database Management Systems, Software Engineering, Cloud Computing, and Object Oriented System Development.