The Scientific Temper (2024) Vol. 15 (1): 1777-1783
E-ISSN: 2231-6396, ISSN: 0976-8653
Doi: 10.58414/SCIENTIFICTEMPER.2024.15.1.28
https://scientifictemper.com/

RESEARCH ARTICLE

Machine learning classifiers to predict the quality of semantic web queries

Gomathi Ramalingam[1*], Logeswari S.[2], M. D. Kumar[3], Manjula Prabakaran[4], Neerav Nishant[5], Syed A. Ahmed[6]

Abstract

In this research, a classification framework to automatically identify well and poorly designed SPARQL queries is proposed. Evaluating SPARQL queries becomes a difficult challenging issue because of the query design and the volume of data to be handled. The proposed context applies various machine learning algorithms including decision trees, k nearest neighbours, support vector machine, and naive Bayes. In addition, two different feature extraction techniques called TFIDF measure and count vectorizer are measured to identify the key features. The experimental results show that the four machine learning classifiers applied are able to classify the SPARQL queries into three categories like well, accepted, and poorly designed queries. It also provides hopeful results with respect to recall, precision, and F1-score. In datasets used for experimentation, it was found that the decision trees classifier outperforms well compared to other classifiers by achieving 92% in terms of F1-measure. Also, the count vectorizer performs well in measuring the TFIDF property to predict the poorly designed queries.

Keywords: SPARQL query, Machine learning algorithms, Classification, Feature selection, Semantic web, Query quality.

Introduction

A database management system (DBMS) is designed to store and retrieve data to the customers as and when necessary, in the required format. The basic mechanism of interaction with customers is in the form of queries. The users of a DBMS always expect to retrieve answers as early as possible (Kamatkar *et al*., 2018). However, many factors become a challenge with respect to the query response time of any DBMS. One of the prominent technologies in the field of computer science is the semantic web (Gupta *et al*., 2022). Normally in semantic web, data will be represented in the form of resource description framework (RDF). And the popularly used language to query the semantic web data is the SPARQL. For better performance of the semantic web data management, we need to use queries which are well designed.

Every query written will be having many different query plans and the process of assessing which plan is the best is not an easy task. Manual assessment of such queries takes much amount of time and so there is a necessity for an automated mechanism to complete this kind of a task. This research work aims to introduce an architecture to identify well-designed, accepted, and poorly-designed SPARQL queries using machine learning algorithms.

The rise of various site pages' step by step prompts the improvement of the semantic web innovation (Gomathi *et al*., 2014). The World Wide Web Consortium (W3C) standard for putting away semantic web information is the resource description framework (RDF). There was research which focusses on the issue of question streamlining of semantic web information. A proficient calculation called adaptive cuckoo search (ACS) for questioning and producing ideal question plan for huge RDF charts was proposed in literature. Tests were led on various datasets with differing number of predicates. The exploratory outcomes have uncovered that the proposed approach has given huge

[1]Bannari Amman Institute of Technology, Sathy, India
[2]Karpagam College of Engineering, Coimbatore, India
[3]Kalasalingam Academy of Research and Education Krishnankoil, India
[4]Madanapalle Institute of Technology and Science, Andhra Pradesh, India
[5]School of Engineering, Babu Banarasi Das University, Lucknow, India
[6]Assistant professor/CSE, Maulana Azad National Urdu University, polytechnic, Hyderabad, India.
***Corresponding Author:** Gomathi Ramalingam, Bannari Amman Institute of Technology, Sathy, India, E-Mail: gomathir@bitsathy.ac.in

How to cite this article: Ramalingam, G., Logeswari, S., Kumar, M. D., Prabakaran, M., Nishant, N., Ahmed, S. A. (2024). Machine learning classifiers to predict the quality of semantic web queries. The Scientific Temper, **15**(1):1777-1783.
Doi: 10.58414/SCIENTIFICTEMPER.2024.15.1.28
Source of support: Nil
Conflict of interest: None.

Received: 03/01/2024 **Accepted:** 06/02/2024 **Published** : 15/03/2024

outcomes as far as question execution time. The degree to which the calculation is productive is tried and the outcomes are recorded. These days graph oriented database engines (GDBMS) are equipped for overseeing enormous volumes of diagram information (Casals *et al*., 2023). These engines, are similar to some other database engine, have an inward question enhancer that proposes a query execution plan (QEP) for each question they process. In research the issue of foreseeing SPARQL question execution was addressed (Hasan *et al*., 2014). AI strategies were used to gain SPARQL question execution from recently executed inquiries. Conventional methodologies for assessing SPARQL inquiry cost depend on insights about the basic information.

As the intricacy of the semantic web increments, productive ways of questioning the semantic web information are turning out to be progressively significant (Hasan, 2014). In addition, buyers of the semantic web information might require clarifications for troubleshooting or understanding the thinking behind delivering the information. Ontologies are the fundamental parts of the semantic web (Lewis *et al*., 2022). Since its starting point and development over the long haul, ontologies have been utilized in different spaces of information. One of the difficulties of dealing with a RDF data set is foreseeing execution of SPARQL questions before they are executed (Zhang *et al*., 2018). With a fast development in the accessible asset portrayal system (RDF) information from unique spaces, the SPARQL question handling with chart structures has become progressively significant (Lin *et al*., 2022). In this pursuit, a two-stage SPARQL question streamlining technique to deal with the SPARQL question was proposed in literature.

This paper is structured as follows: In section 2, the methodology of predicting the quality of the SPARQL query is described. In section 3, the datasets used and the output of applying machine learning algorithms for prediction is discussed. In section 4, the results and discussion are presented. In section 5, the conclusion summarizes the key results of the research.

Materials and Methods

The components of the proposed architecture is explained in this section to classify the SPARQL: queries. The architecture consists of four main components:

- Manual SPARQL query labeling
- Query pre-processing
- Training the model
- Prediction & evaluation

The steps involved in each of these steps are illustrated in Figure 1.

Manual SPARQL Query Labeling

The technique used in this research to find the labeled data is the Delphi approach (Williams *et al*., 1994) which

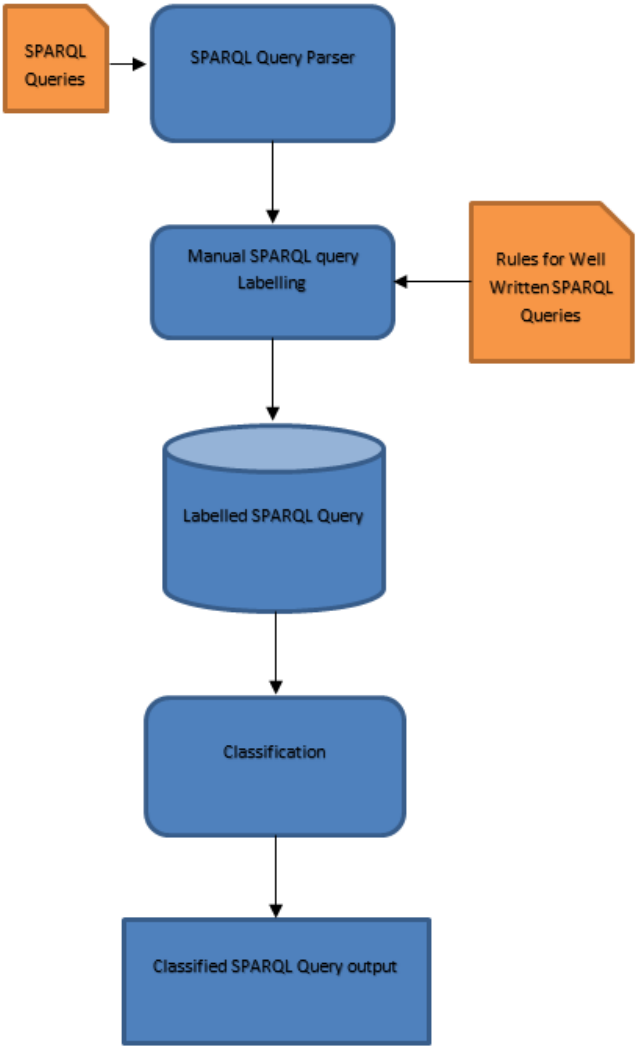

Figure 1: Query classification architecture

manually classifies the SPARQL queries into two categories: Well and poor. In continuation, the poor category queries are further classified as two varieties: Poor and accepted. The Delphi approach is used for organizing group communication, apprehending judgments from areas with imperfect research indication, and determining differences of views. The group for this research comprises of ten university Professors, database developers, and database administrators. In the first iteration, the SPARQL queries from the datasets were given to the experts group, and are asked to classify them into the given categories. The data from the first iteration was collected, analysed, and further reduced by defining the SPARQL queries that has a common class from the maximum of the 30 experts. Finally, the label for these determined SPARQL queries is determined and they were excluded from the next iteration.

In the second iteration, the remaining SPARQL queries were given to the same experts group with necessary data about the responses from the first iteration. The experts are asked to reclassify each query based on the given data.

The data from the second iteration is collected, and the common class for each SPARQL query is decided as the final label. In the third iteration questionnaire responses from the previous questionnaire were provided for the SPARQL queries. The expert's group will again rate the final label from the first and second iterations on a 5-point Likert scale. The SPARQL queries with an average label of less than 3 were omitted from the dataset. An example of a poorly written SPARQL query is given below:

```
SELECT * WHERE
 {
 ?deal a :Deal ;
 :employee ?employee ;
 :customer ?customer .
     ?employee :name ?employeeName ;
 :involvedAsEmployee ?matter .
     ?customer :name ?customerName ;
 :involvedAsCustomer ?matter .
}
```

The query is graphically represented as below in Figure 2:

When we try to analyse this query, iit is found that the : name relation is practical, which means the joins with the patterns ?employee :name ?employeeName and ?customer :name ?custName should not create misfortune. The important planning query here is which join on the ?deal or the ?matter should be evaluated first? The response depends on the data: if the deal join is selective, it should be done first, or if less people are involved in the same stuff, the matter join should be done first. This is the point where the decision-making of the optimizer comes into picture. In the ideal world, it'll choose based on data and will decide correctly. But in some situations, like outdated data or multifaceted inter-dependencies in the data, there is a possibility for the optimizer to make a wrong choice. So, it's likely that there are many deals and the optimizer decides to first identify people which were involved in the same matter, but it turns out that some matters in the past involved many people. So that join is not selective and the optimizer ended with a wrong choice.

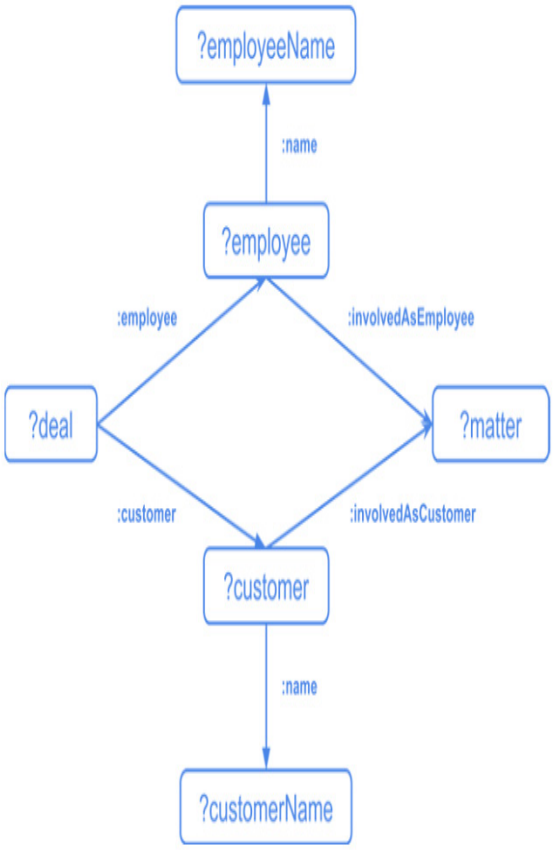

Figure 2: Graphical representations of the SPARQL query

Pre-Processing Phase

Data pre-processing is an important step in any data processing mechanism, since it can improve classification results in a few cases (Kumari *et al*., 2017). Data pre-processing includes steps like input data cleaning, in which the SPARQL queries with syntax errors are eliminated. To identify such errors, the query parser is used. The next step is tokenization, in which involves the SPARQL query statements are broken down into words, symbols, or elements called tokens. This process is vital since it extract the relevant features from the SPARQL queries.

Feature Extraction

Feature extraction is a step used to extract the informative set of features in order to increase the classifier accuracy. Every SPARQL query is split into tokens in the previous step and identifying which terms in the SPARQL query are most distinguishing for that query can be considered an enlightening feature. Machines cannot process data in its raw form. In order to make the text comprehensible to machines, it should be converted into a format that can be easily interpreted by computers. The popular mechanisms to do this conversion is the term frequency-inverse document frequency (TF-IDF) and bag-of-words (BoW) which converts text into numerical vectors. For extracting query features, both TF-IDF and BoW can be used. The TF-IDF consists of two parts in which TF guesses how important a term is in a SPARQL query. The more existence of a term in a query means the more significant it is.

The term frequency of each term tk of any query q is computed as below equation 1:

$$TFtk, q = ftk, q/maxfq \quad (1)$$

where $maxfq$ indicates the maximum term frequency of all terms that is present in the query.

The second IDF is calculated by using the total number of queries in the corpus and divided by the number of queries where the term appears as in equation 2.

$$IDFt = log \frac{|Q|}{|Qt|} \quad (2)$$

Where $|Q|$ is the total number of queries, and $|Qt|$ is the number of queries where the term t appears.

Table 1: Summary of dataset statistics

Name of the dataset	*Total number of queries taken for experimentation*	*Number of parsable queries*	*Number of distinct query strings*	*Correct query*
LUBM	220	215	145	123
CIA World Factbook	240	232	156	148

The result of this step is a vector of integers presenting the TF-IDF of each query. Reasonably, the BoW approach also well-known as count vectorizer considers each word count as a feature. Bags-of Words represents text that illustrate the occurrence of words within a text. It uses two information: first, a vocabulary of known words, and second, a measure of the count of their presence.

Classification

The job of predicting a class label for input is fundamentally a classification problem. In this method, the SPARQL queries are divided into three categories: well-designed queries, accepted queries, and poorly designed queries. The classification process is done in two phases: The training phase and the prediction phase. In the training phase, the data is decomposed into a set of features based on feature generation models, which includes the vector space model for textual data. As stated earlier, the classification approach is regarded as a modeling problem where a class label is predicated based on a given example of input data. Class labels are usually string values like "well-designed" "poorly-designed," and "accepted". These ordinal data labels should be encoded to a sole ordinal integer value, like "well-designed" = 2, "accepted "= 1, and "poorly-designed" = 0 before given to the classifier. In this situation, a binary classifier is needed to perform this task. The classifier is processed to predict a discrete probability distribution of a query belonging to a specific class. In this research, the most popular machine learning classification algorithms which includes logistic regression, k nearest neighbours, decision trees, support vector machine, and naive Bayes (Tripathy *et al*., 2016) are separately applied to the proposed query classification architecture.

Evaluation

To evaluate the performance of the system, the standard evaluation metrics like precision, recall, and F-score are used. The formula for these metrics is given below in equation 3,4 and 5:

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

$$Recall = \frac{TP}{TP+FN} \quad (4)$$

$$F1 - Score = 2 * \frac{Recall*Precision}{Recall+Precision} \quad (5)$$

Where P is the positive predicted value
N is the negative predicted value
TP indicates the true positive predicted value
TN indicates the true negative predicted value
FP indicates the false positive predicted value
and FN indicates the false negative predicted value.

Observation/Results

In this section, experiments are carried out to evaluate the classification performance of several machine learning algorithms in classifying the SPARQL queries as well-designed or poorly-designed and to predict the most appropriate feature for classification. The analysis of this investigation will deliver understanding into the efficiency of different machine learning algorithms and feature extraction techniques used to perform this task.

Dataset

Our evaluation is based on two different datasets, namely, Leigh University benchmark (LUBM) dataset, and the Central Intelligence Agency (CIA) World Factbook dataset. The LUBM benchmark is intended to assess the performance of those repositories with respect to extensional queries over a huge data set that obligates to a single truthful ontology. It comprises of university field ontology, a set of test queries, and several evaluation metrics. The dataset used for this research is LUBM5 which contains about 645,649 triples. The CIA World Factbook provides us with data about 250 countries defined using RDF triples. Data about government, people, transportation, and many more are present in this dataset. It consists of more than 100,000 RDF statements.

The following Table 1 shows the statistics about the datasets.

The following Table 2 describes a summary of dataset information after binary manual labeling.

The following Table 3 describes a summary of dataset information after binary manual labeling.

Experiment Setting

To assess the performance of the machine learning classifiers, k-fold cross-validation method is applied (Zhang

Table 2: Summary of dataset statistics after binary manual labeling

Dataset	*Correct query*	*Well designed*	*Poorly designed*
LUBM	123	85	38
CIA World Factbook	148	121	27

Table 3: Summary of dataset statistics after ternary manual labeling

Dataset	*Correct query*	*Well designed*	*Accepted*	*Poorly designed*
LUBM	123	42	54	27
CIA World Factbook	148	65	52	31

Table 4: Results of LUBM dataset

		TF-IDF		*Count vectorizer*	
		Poorly designed	*Well designed*	*Poorly designed*	*Well designed*
DTC	Precision	0.72	0.90	0.72	0.90
	Recall	0.72	0.90	0.75	0.90
	F1-Score	0.72	0.90	0.76	0.90
KNN	Precision	0.66	0.82	0.73	0.88
	Recall	0.43	0.91	0.65	0.91
	F1-Score	0.52	0.85	0.71	0.90
SVM	Precision	0.95	0.85	0.78	0.90
	Recall	0.52	0.98	0.71	0.92
	F1-Score	0.71	0.90	0.74	0.89
NB	Precision	0.37	0.86	0.34	0.86
	Recall	0.65	0.57	0.74	0.56
	F1-Score	0.45	0.71	0.50	0.68

Table 5: Results of CIA World Factbook dataset

		TF-IDF		*Count vectorizer*	
		Poorly designed	*Well designed*	*Poorly designed*	*Well designed*
DTC	Precision	0.74	0.88	0.71	0.89
	Recall	0.74	0.88	0.74	0.89
	F1-Score	0.73	0.87	0.71	0.88
KNN	Precision	0.62	0.77	0.70	0.82
	Recall	0.40	0.81	0.62	0.89
	F1-Score	0.51	0.80	0.69	0.89
SVM	Precision	0.94	0.81	0.73	0.89
	Recall	0.51	0.92	0.70	0.88
	F1-Score	0.68	0.91	0.73	0.82
NB	Precision	0.33	0.83	0.32	0.84
	Recall	0.56	0.54	0.77	0.53
	F1-Score	0.42	0.69	0.49	0.62

et al., 2018). In the first step, the dataset is divided based on k=5 subsets. Each subset is again divided into two subsets: one for training and the other for testing. About 80% of datasets are used for training and 20% is used for testing. The purpose of training dataset is to make the classification model understand the problem and the purpose of testing set is to evaluate the fitness of the classification model used. Experiments are conducted in two sets: One set for two class labels (Well designed and poorly designed) and another for three class labels (Well designed, Accepted, and Poorly designed). Each set of experiments was executed on the two datasets. The experimental results for four machine learning classifiers were compared, and the efficiency of each classifier was assessed based on the metrics precision, recall, and F1-score. The experiments use two features: TFIDF features and BoW features.

Discussions

The results of classification accuracy using the evolution metrics recall, precision, and F1-measure is presented in this section. For the classification of the datasets, the popular machine learning classifiers including decision trees (DTC), k nearest neighbours (KNN), support vector machine (SVM), and naive Bayes (NB) were trained. The classification results using two class labels (Poorly designed, well designed) for the LUBM dataset are shown in Table 4 and the results for the CIA World Factbook dataset are shown in Table 5.

When we compare the performance of the classifiers with TFIDF and BoW vectorizer features with respect to recall, DTC plays well in classifying queries into poorly designed and well-designed categories. SVM is better than the classifiers. However, when we consider the accuracy

Table 6: Results of LUBM dataset

		TF-IDF			*Count vectorizer*		
		Poorly designed	*Accepted*	*Well designed*	*Poorly designed*	*Accepted*	*Well designed*
DTC	Precision	0.70	0.87	0.88	0.69	0.87	0.88
	Recall	0.70	0.82	0.88	0.71	0.87	0.88
	F1-Score	0.70	0.86	0.87	0.72	0.86	0.87
KNN	Precision	0.62	0.76	0.81	0.71	0.83	0.84
	Recall	0.42	0.82	0.89	0.62	0.91	0.92
	F1-Score	0.51	0.78	0.83	0.70	0.89	0.88
SVM	Precision	0.91	0.77	0.82	0.74	0.89	0.88
	Recall	0.50	0.90	0.94	0.68	0.86	0.87
	F1-Score	0.68	0.81	0.85	0.72	0.83	0.84
NB	Precision	0.32	0.78	0.81	0.32	0.80	0.81
	Recall	0.63	0.52	0.54	0.72	0.51	0.52
	F1-Score	0.41	0.68	0.72	0.49	0.66	0.65

Table 7: Results of CIA World Factbook dataset

		TF-IDF			*Count vectorizer*		
		Poorly designed	*Accepted*	*Well designed*	*Poorly designed*	*Accepted*	*Well designed*
DTC	Precision	0.72	0.88	0.87	0.72	0.87	0.88
	Recall	0.72	0.88	0.87	0.71	0.87	0.88
	F1-Score	0.70	0.87	0.86	0.69	0.87	0.88
KNN	Precision	0.58	0.72	0.74	0.69	0.80	0.81
	Recall	0.38	0.74	0.76	0.60	0.80	0.82
	F1-Score	0.49	0.74	0.77	0.64	0.82	0.82
SVM	Precision	0.92	0.73	0.74	0.71	0.82	0.82
	Recall	0.49	0.86	0.88	0.68	0.81	0.83
	F1-Score	0.64	0.84	0.88	0.69	0.82	0.82
NB	Precision	0.31	0.79	0.80	0.31	0.81	0.83
	Recall	0.52	0.48	0.51	0.74	0.50	0.51
	F1-Score	0.40	0.62	0.63	0.47	0.54	0.57

metrics this is not the actual case. When considering the weighting accuracy in terms of F-score, both DTC and SVM achieved the best result in each dataset.

The classification using three class labels for LUBM and CIA World Factbook dataset is presented in Tables 6 and Table 7 respectively.

The experimental comparisons and values indicate that the decision tree classifier with count vectorizer features performs well compared to the other classifiers together in two-class as well as three-class classifiers for both datasets. The obtained hopeful results specify that the proposed architecture is efficiently able to perceive poorly designed SPARQL query. This research helps the other researchers, database developers, and SQL programmers in the same field of research to use the architecture as a tool to detect the poorly designed SPARQL queries. Moreover, the proposed architecture serves as an important component of optimizing the performance of SPARQL query.

Conclusion

Information retrieval from the semantic web data highly relies on the SPARQL queries written and its quality. However, poorly designed SPARQL queries leads to performance problems. In this research, we proposed an architecture for evaluating quality of a SPARQL query using machine learning algorithms. The effectiveness of applying various machine learning classifiers in the classification of SPARQL queries into three categories: Well-designed, accepted, and poorly designed was experimented. The machine learning classifiers used includes k- nearest neighbours, decision trees, support vector machine, and naive Bayes. Features, such as term frequency, inverse document frequency, and bag-of-words count vector, are used in the training and testing of the classifiers. The results of the experiments, conducted on two different datasets, shows that the decision trees and support vector machine classifiers produces promising results in the classification of SPARQL queries based on metrics. Future research can be performed focussing on exploring different features of SPARQL queries to progress classification accuracy. Also, the proposed architecture may be tested across different RDF datasets.

Acknowledgement

The authors would like to thank the management for providing the required resources to complete this research work.

References

Casals, D., Buil-Aranda, C., & Valle, C.(2023). SPARQL query execution time prediction using Deep Learning.

Gomathi, R., & Sharmila, D. (2014). A novel adaptive cuckoo search for optimal query plan generation. The Scientific World Journal,2014.

Gupta, R., & Malik, S. K. (2022, December). Visualizing Semantic Web Data using Various Tools Focusing RDF, OWL and SPARQL. In 2022 11th International Conference on System Modeling & Advancement in Research Trends (SMART),1456-1460. IEEE.

Hasan, R. (2014). Predicting SPARQL query performance and explaining linked data. In The Semantic Web: Trends and Challenges: 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings 11,795-805. Springer International Publishing.

Hasan, R., & Gandon, F. (2014, August). A machine learning approach to sparql query performance prediction. In 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 1, 266-273. IEEE.

Kamatkar, S. J., Kamble, A., Viloria, A., Hernández-Fernandez, L., & Cali, E. G. (2018). Database performance tuning and query optimization. In Data Mining and Big Data: Third International Conference, DMBD 2018, Shanghai, China, June 17–22, 2018, Proceedings 3,3-11. Springer International Publishing.

Kumari, R., & Srivastava, S. K. (2017). Machine learning: A review

on binary classification. International Journal of Computer Applications, 160(7).

Lewis, C. N., Cabrera, V. L., & De Castillo, I. (2022, September). Semantic Web and Ontologies to Preserve Gastronomic Knowledge. In 2022 V Congreso Internacional en Inteligencia Ambiental, Ingeniería de Software y Salud Electrónica y Móvil (AmITIC) (pp. 1-7). IEEE.

Lin, X., & Jiang, D. (2022). A Two-Phase Method for Optimization of the SPARQL Query. Journal of Sensors, 2022.

Tripathy, A., Agrawal, A., & Rath, S. K. (2016). Classification of sentiment reviews using n-gram machine learning approach. Expert Systems with Applications, 57, 117-126.

Williams, P. L., & Webb, C. (1994). The Delphi technique: a methodological discussion. Journal of advanced nursing, 19(1), 180-186.

Zhang, W. E., Sheng, Q. Z., Qin, Y., Taylor, K., & Yao, L. (2018). Learning-based SPARQL query performance modeling and prediction. world wide web, 21, 1015-1035.