Journal of Computers, Mechanical and Management

Journal homepage: www.jcmm.co.in

Volume 1 Issue 1

Astoundingly Smart System Furnishing Ranking of Big Data In Search Engines

Kakoli Banerjee*and Shishir Dua

Department of Computer Science and Engineering, JSS Academy of Technical Education, Noida, Uttar Pradesh, India 201301

Abstract

The abruptly escalating internet is sensational. It inculcates a humungous volume of big data, which is obsolete and tedious to manage, scrutinize, analyze and perform operations upon them in conventional ways. Big data has thus expedited the search and retrieval of information, necessitating the development of contemporary search algorithms to aid this process. However, the primary hindrance for the first and second generations of conventional search engines was the syntax of keywords devoid of semantic meaning and the lack of a knowledge base that linked disparate web material. This article presents a framework based on trendy technologies, specifically Extracting, Transforming and Integrating (ETI) processes, ontology graphs, and indexing Resource Description Framework (RDF) using the wide-column Not only Structured Query Language (NoSQL) method. The most significant contribution in this regard is developing a mathematical model to compute the similarity score between a query and stored RDF documents using semantic relations. Numerous operations were carried out to evaluate the effectiveness of the proposed methodology in installing data sources, such as DBpedia and YAGO dataset. Insofar as experimental results are concerned, the suggested model achieves greater precision than other comparable systems.

Keywords: YAGO Dataset; DBpedia; Ontology; Semantic Web; NoSQL

1 Introduction

Search engines underwent a decent frequency of generation versions. In the initial generation, it thoroughly relied on depicting information and corresponding data which matched the input query only. Contemporary second-generation search engines enable real-time query scrutiny and analysis by employing machine learning techniques, deep learning algorithms and complementary data models, subsequently deducing the worthwhile keywords of query analysis and ditching semantic stuff [1]. Nowadays, these are inculcating consideration of part of tremendous enterprises to endow with access to information. The internet is the most substantial syntax source of data, datasets and information ever initiated. Static web, including Hyper-Text Markup Language (HTML), elucidates the syntax structure of information. Henceforth, a traditional search engine can comprehend syntax but cannot infer the speculations and requirements. Figure 1 depicts the third generation of search engines that utilize semantic relations amongst immense things to evolve a knowledge graph [2]. Ontology validates the elements that exist or may have existed in any field or context and is continually applied to illustrate semantic relations. The ontology classes elucidate the verbs, word senses, or notion and association types. Efficacious tools for a semantic search engine employing ontology include Ontology Web Language (OWL) and Resource Description Framework (RDF), evolving decipherable information. RDF is a language for interpreting information about sources accessible on the internet [3–5]. In RDF, the predicate denotes relationships between the subject and the object. The object is rather the value. It includes another resource or a literal value comprising a number or word.

*Corresponding author: kakoli.banerjee@jssaten.ac.in
Received: 28 August 2022; **Accepted:** 17 September 2022; **Published:** 30 October 2022
© 2022 Journal of Computers, Mechanical and Management.
This is an open access article and is licensed under a Creative Commons Attribution-Non Commercial 4.0 International License.
DOI: 10.57159/gadl.jcmm.1.1.23018.

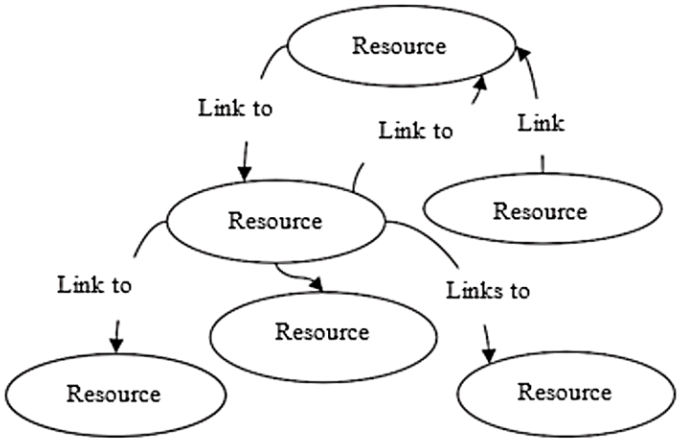

Figure 1: The general structure of the semantic web.

There are quite a few operational obstructions and research highlights related to search engines, which are enlisted in this paragraph. These incorporate [6–8]:

- The current well-scaled traditional or standard search engines that pivot on syntax relationships between keywords or topics, ditching semantic relations.
- The semantic search engines that do not scale decently and focus on the maiden field in the search area and context
- The big data possess predicament, namely time complexity, scalability, and availability required to enhance the knowledge base of any search engine

The contributions cited in this article are designated into three paramount categories. These are furnished further on (a) recommendation of semantic search system framework, (b)augmentation and enhancement of knowledge base through more than one source using proposed Extracting, Transforming and Integrating (ETI) processes, (c) proposal and evaluation of a novel mathematical representation for computing and determining semantic relations between distinct subjects, and (d) variety of testing factors, held for evaluating recommended framework.

2 Background

This section includes a concise review and summary of a simple survey of the most common forms of Information Retrieval (IR) models. Also, different conventional methods of measuring the degree of similarity at the level of characters or words are briefly reviewed.

2.1 Different models of information retrieval systems

An IR model governs how a document and a query are represented. There are two common main models: Boolean and vector space models, [6] of which the vector space model is employed in the present work.

Boolean model

This model is the simplest and oldest model used in search engines. The concept of exact match and the laws of Boolean algebra is used to match the search results with the query entered by the user [6]. The weight of keyword k_i in page p_j is calculated according to Eq. [1]

$$w_{ij} = \begin{cases} 1 & \text{if } k_i \text{ appears in } p_i \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Vector space model

In this model, a document is represented as a weight vector. The weight of keyword k_i in page p_j is no longer 0,1 as in the previous model but can be any number. We can use Keyword Frequency - Inverse Page Frequency (KF-IPF) scheme to assign a weight for each component.

Let M be the total number of pages in the system and pf_i be the number of pages in which keyword k_i appears at least once. Let f_{ij} be the frequency count of keyword k_i in page p_j [6]. Then keyword frequency kf_{ij} is given by Eq. 3. The inverse document frequency ipf_{ij} is given by the Eq. [2]. Finally, we can get the component's weight using Eq. 4.

$$kf_{ij} = \frac{f_{ij}}{\max(f_1, f_2, \ldots, f_m)} \tag{2}$$

$$ipf_{ij} = \log\left(\frac{M}{pf_i}\right) \tag{3}$$

$$w_{ij} = kf_{ij} \times ipf_{ij} \tag{4}$$

The Needleman-Wunsch method is a dynamic model commonly used in biology and bioinformatics to compare genome sequencing, as it has recently been used in some standard search engines. DNA consists of a large sequence of a specific set of string characters, where this method measures the degree of similarity by aligning two entire sequences. The usage of this method can be logical and correct if the two sequences are of the same length and share a high degree of similarity [8]. The Smith-Waterman method is another example of a dynamic programming model and is also used in bioinformatics, which can compare two sequences by aligning each character with the other to obtain the highest degree of similarity. This method does not require that the two sequences have the same length and have a high degree of similarity. However, the good thing about this method is that one can extract similar areas in any two sequences and do not necessarily have a complete similarity [9]. Longest Common SubString (LCS) method is a sequential comparison between two sequences, and the degree of similarity depends on the length of the similar contiguous chain of characters. That is, the longer the length of the string containing similar characters, the greater the proportion of similarity, even if the meaning is different. Unlike the last two methods, LCS calculates the similarity between the two sequences based on the number of operations that match them. The operations, such as deletions, substitutions, or insertions, are calculated on a sequence of characters. The degree of similarity here reflects the distance between the two sequences. This method is used in some search engines to fix errors in the user-submitted query following one of the terms in the utilized database [2, 3].

2.2 Term-based similarity measurement methods

The cosine method is a similarity score mathematical measurement between two vectors that calculates the cosine of the angle between them. The weight of each Wikipedia article is calculated by this method. The Euclidean distance (ED) method is the distance measurement between two strings or two vectors. The similarity score between two elements is one minus the distance of these two elements. Distance is calculated as the square root of the sum of squared differences between related elements. Dice's Coefficient is twice the amount of common terms in the associated strings divided by the total number of words in both strands [10–12].

2.3 Semantic-based similarity measurement methods

The Explicit Semantic Analysis (ESA) method is a measure used to estimate the semantic relatedness among two arbitrary subjects. The relatedness of pair documents in the same language is evaluated by this method. The latent Semantic Analysis (LSA) method is popular for estimating semantic-based similarity. LSA assumes that strings are close in meaning and will happen in similar parts of the text. LSA includes a matrix that holds word counts per paragraph. In the matrix, rows describe distinct words, and columns represent a specific paragraph. Second-order Occurrence Pointwise Mutual Information (SOC-PMI) is a semantic score measurement method using pointwise information to sort lists of significant neighbor terms of two strings. The relation score between two strings that do not occur regularly can be calculated.

3 Related Works

Standard search systems include search engines like Google, Yahoo, and Bing, directories like Deutsche Medizinische Online Zeitung (DMOZ), and Meta Search systems like Dogpile and Mamma. The vast majority of common search techniques are extremely popular, yet their results are occasionally erroneous, with low precision and high recall. Modern and intelligent search systems, such as Swoogle, Semantic Web Search Engine (SWSE), and Falcons Object Search [12], are created with the semantic approach in mind. Swoogle is a system that crawls and indexes web documents based on their semantic content. It includes four major components: data finding, metadata development, data analysis, and data retrieval. It permits the classification of metadata using the rational surfer model. However, these systems had problems, such as insufficient indexing of enormously big data and slow query response [13]. Moreover, Swoogle lacks a technique for sorting relevant results by relevance. Hogan et al. [14] have referred to the semantic approach as SWSE. Based on RDF and link-related data, it is a comprehensive search system that offers comparable services to conventional search engines. It includes the crawling, indexing, ranking, argumentation, and retrieval phases.

However, SWSE has some flaws, such as

- The system does not scale well.
- The system does not appear resistant in the face of varied, noisy, inconsistent, and possibly contradicting data obtained.

Hakia is an additional system that functions as a comprehensive semantic engine for particular applications. This system is a different form of search engine that produces more accurate and trustworthy results than standard search engines.

It includes a query processor, ontology analyzer, QDex store, and ranking methodology. This system, like others, relies on correlating outcomes in terms of meaning as opposed to statistical methodologies, which lends strength and credibility to the results. Fatima et al. [15] developed a system reliant on the semantic web. For sorting and retrieving data, the suggested search engine in this system depended on a robust query language processor and an intuitive user interface. This system lacks novel algorithms that increase system performance, regardless of the technical methods employed.

4 Method

4.1 Proposed framework

The suggested framework, Semantic Engine based on Enhancing Knowledge (SEEK), is designed modularly and reasonably composed of two separate stages. Firstly, the offline stage is a back end where the server runs solely away from users. The offline phase involves ETI and indexing based on semantic relation processes. Secondly, the knowledge source phase converts JavaScript Object Notation (JSON) format into RDF schema and stores related subjects into a cache table for faster retrieval. Thirdly, the online phase is a front-end stage where a user can operate directly with the knowledge source in real time. As presented in Figure 2, Each bold box focuses on the contributions of the present work.

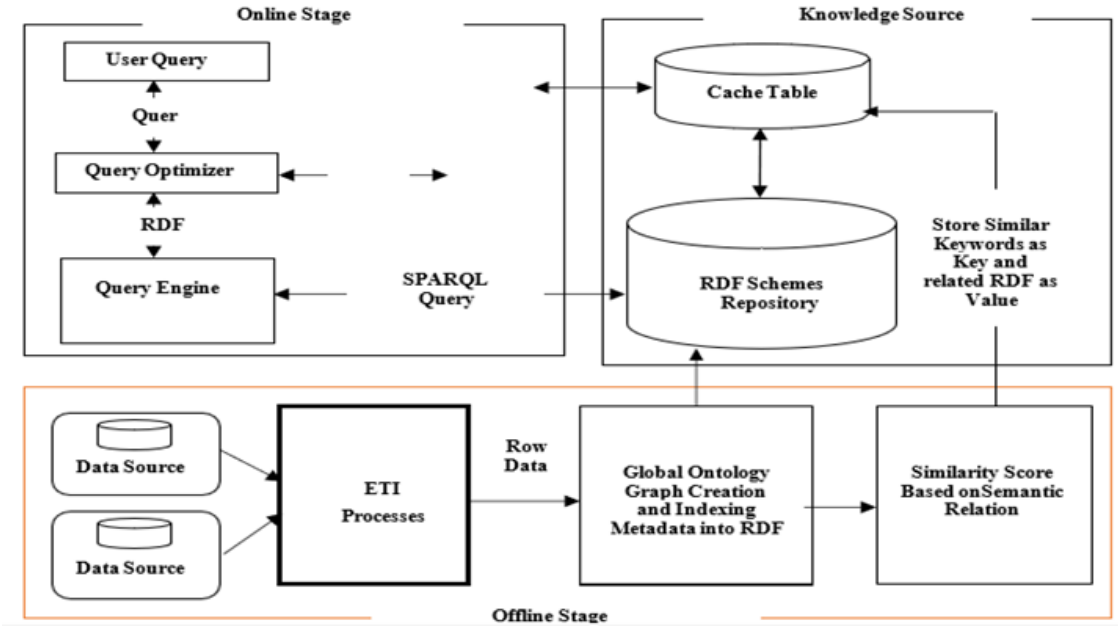

Figure 2: Proposed SEEK architecture.

Offline phase

This phase consists of three integrated stages: ETI (Extracting, Transforming, and Integrating) processes, indexing process, and similarity score calculation process based on semantic relation, of which the present study deals with the ETI and the similarity score calculation processes.

ETI Processes: This stage is responsible for ETI processes that extract data from different data sources and convert it into uniform JSON format. The ETI job manager on a master thread invokes a schema analyzer to identify the schema of the API data sources (e.g., Facebook, Twitter, DBpedia APIs). This stage outputs a uniformly integrated schema in JSON format. ETI processes are initiated on the master thread and distributed on several slave threads using the MapReduce technique for parallel processing. The extracting process involves fetching data from the source, which should be correct and accurate. The transforming process performs a series of complicated data cleaning and converts it into JSON format. The integrating process is used to bind extracted data into another JSON format. Algorithm 1 represents the overall steps of how the schema is being created. In the schema creator procedure, the master thread parallelly assigns collections of different data sources (input) to slave threads for creating the schema (output).

Algorithm 1 Schema Creator Procedure

```
procedure SCHEMA CREATOR
    Input: Data Source (DS_i)
    Output: Schema
    Begin
    dataset ← DS_i.open connection
    count ← dataset.collection.Names()
    Schema ← null
    for i ← 0 to count do
        for record ∈ dataset.collection(i) do
            for column_name ∈ record do
                Schema ← schema ∪ column_name
            end for
        end for
    end for
    Return Schema
end procedure
```

After that, Algorithm 2 shows how data can be extracted. In the extraction procedure, the inputs are the data source, the number of threads assigned to this data source (thread counter) and the schema design from the previous method. The output of this procedure is Row data in batches.

Algorithm 2 Extraction Procedure

```
procedure EXTRACTION
    Input: Data Source (DS_i), thread counter (TC), Schema
    Output: Row Data
    Begin
    dataset ← DS_i.open connection
    count ← dataset.collection.Names()
    Limit ← Count
    for i ← 0 to count do
        Start ← limit ∗ i
        Batch ← Read data (batch size)
        Row Data ← Extract (process id, Batch, DS_i, start, limit, schema)
    end for
    Return Row Data
end procedure
```

Finally, the transformation procedure can record all the distinctions in the data source and compile the schema into a JSON format, as shown in Algorithm 3. In the transformation procedure, the inputs are the data source, extracted row data, and schema design. The output of this procedure is integrated data in JSON format.

Algorithm 3 Transformation Procedure

```
procedure TRANSFORMATION
    Input: Data Source (DS_i), Row Data, Schema
    Output: Data in JSON format
    Begin
    Initialize schema ()
    Data Collection ← Row Data.Length
    Limit ← Data Collection.Count
    Start ← 1
    while Start < Limit do
        Clean (Row Data)
        Remove Duplicates (Row Data)
        Schema ← Bind Row Data
        Start++
    end while
    JSON ← Schema
    Return JSON
end procedure
```

Similarity score calculation processes: A basic structure of the semantic ontology graph is depicted in Figure 3.

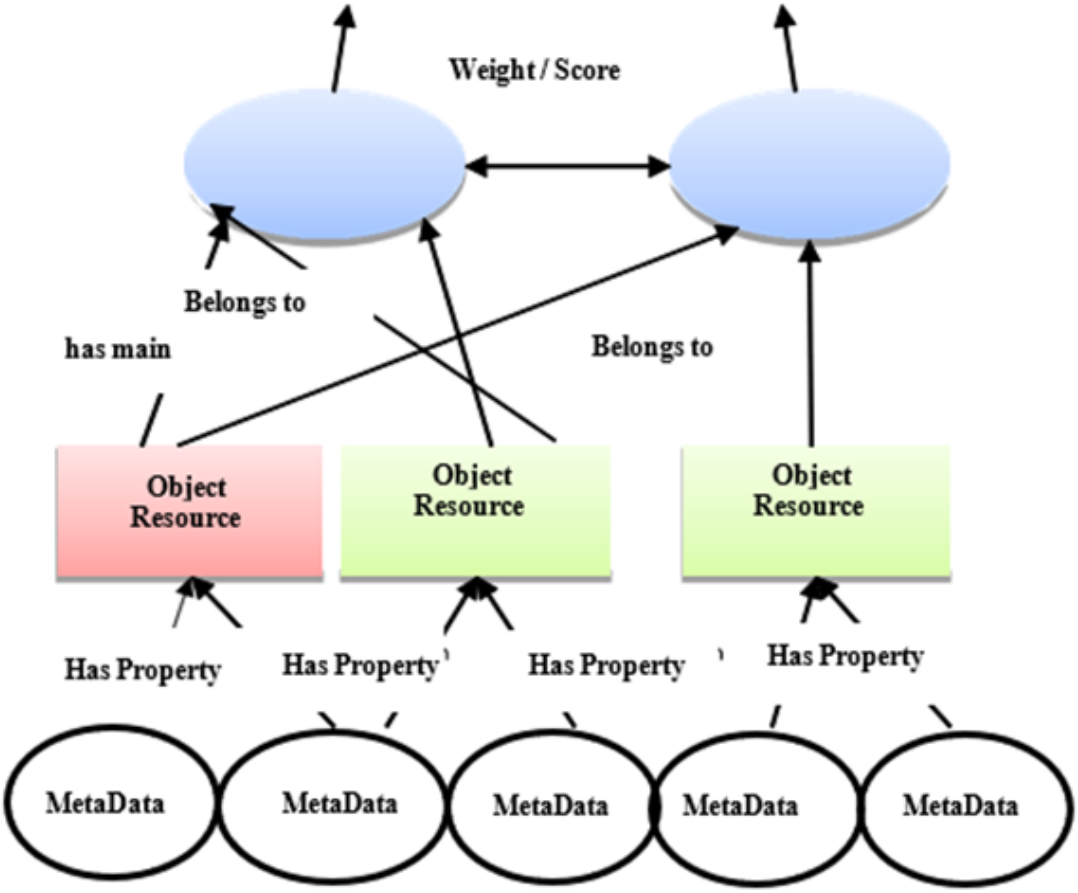

Figure 3: Structure of the Semantic Ontology Graph [8].

Semantic Score (SS) equation, represented by Eqs. [5] and [6], is used to measure the weighting score between two subjects (I and J) in the ontology graph [8], where:

- S_i is the count of input relations of the subject I from different objects.
- S_j is the count of input relations of subject J from different objects.
- T is the total number of subjects in the graph.

$$SS_{in}(S_i, S_j) = 1 - \frac{\max(\log S_i, \log S_j) - \log(S_i \cap S_j)}{\log T - \min(\log S_i, \log S_j)} \tag{5}$$

$$SS_{in}(S_i, S_j) = \begin{cases} 0 & S_i \neq S_j \\ 1 & S_i = S_j \\ \text{otherwise} & S_i \text{ related to } S_j \end{cases} \tag{6}$$

Figure 4 illustrates an example of calculating semantic scores between two subjects (I and J). If the subject (I) has four incoming links from different objects (A, D, E, and F). Then the semantic relation of the subject (I) would be four. Considering that the subject (J) has five incoming links from different objects (A, B, C, F, and G), then the semantic relation of the subject (J) would be five. However, two objects (A and F) belonged to both subjects. So, the semantic relation of both subjects (I and J) is two. If the ontology graph has 1000 subjects, then the semantic score by applying Eq. [5] equals 0.83, which means there exists high semantic relation between two subjects (I and J) as per Eq. [6].

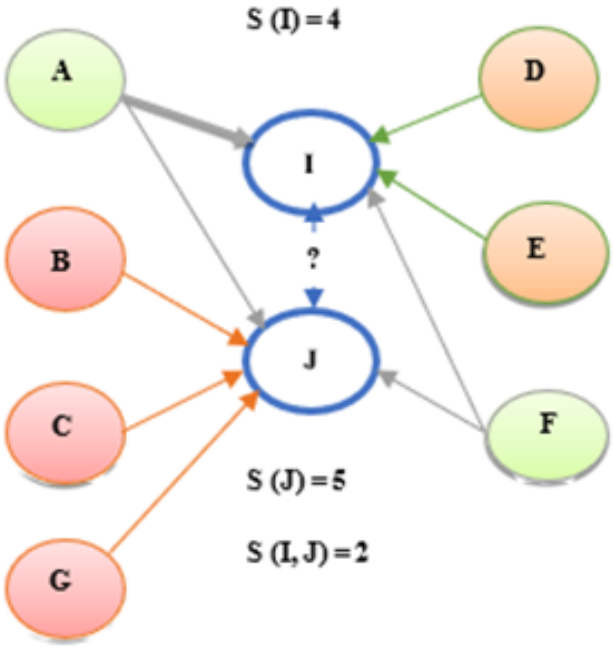

Figure 4: Example of calculating Semantic Score between two subjects (I and J).

Knowledge source phase

The present work employs HBase integrated with Hadoop for knowledge sourcing. It consists of one master and six slave machines. The integration of HBase with Hadoop archives has many advantages as (a) Hadoop runs batch jobs using the MAP - Reduce technique on distributed machines to achieve maximum throughput and overall performance [16, 17], (b) HBase provides good latency, scalability and fault tolerance wherein the replication process handles fault tolerance. Scalability is handled by splitting each table horizontally into regions, and (c) The integration is done successfully because they use the same underlying HDFS file systems [18].

Online phase

This phase incorporates two crucial stages. The initial stage instills optimizing input query, and the consequent stage is the actual retrieving process by the query engine. The query optimizer phase is implemented wherein the stop words removal, stemming and keyword expansion processes are applied to the input query from the user. Wordnet tool is used to expand keywords to extend search domains. The semantic score between extended keywords and actual query keywords is computed to generate a key of the input subject. Subsequently, the value of the corresponding key is returned from caching table. Then the query engine is implemented. At this stage, the actual retrieval process is applied using the Simple Protocol and RDF Query Language (SPARQL) query to return the objects of the corresponding subject value. The query engine is used to generate a SPARQL query to retrieve related information from RDF and the corresponding source.

4.2 Experimental setup

The proposed framework is evaluated using a cluster of, one head machine node and six slave machines. The head machine had Intel core i7 processor @2.6GHz with 16GB RAM. The slave machines possessed Intel core i3 processor@ 2.2GHz with 4GB RAM for each. Hadoop version 2.9.2 was used as an open-source tool. Hadoop was assisted with three workers per MAP operation and two workers per REDUCE operation. Also, Apache HBase version 2.2.0 was utilized. Multiple parallel slave workers were considered for the execution of ETI jobs.

4.3 Datasets description

The proposed framework was evaluated using two real-world datasets. Table 1 depicts some statistical information from these datasets. The two real-world datasets are:

- DBpedia, a pre-eminent real-world dataset utilized in the semantic web research community. The DBpedia version used to compute and augment the acuteness of cited framework is 2016-04.
- YAGO, yet another great real-world dataset developed at Max Planck institute. The YAGO version used to evaluate our framework is 2015-03.

Table 1: Brief statistical information of DBpedia and YAGO datasets.

Parameters	DBpedia	YAGO
# Entities in person Domain	1.5 M	1.3 M
# Entities in work Domain	490 K	510 K
# Entities in the organization Domain	275 K	350 K
# Entities in places Domain	810 K	1.2 M
# Entities in the biology Domain	301 K	150 K
# N-Quads used	150 M	120 M

5 Results and Discussion

5.1 ETI execution results

Figure 5 depicts the average ETI execution time recorded for different workers. According to this figure, seven threads were utilized owing to the reason that they stipulated pretty slightest of the execution time.

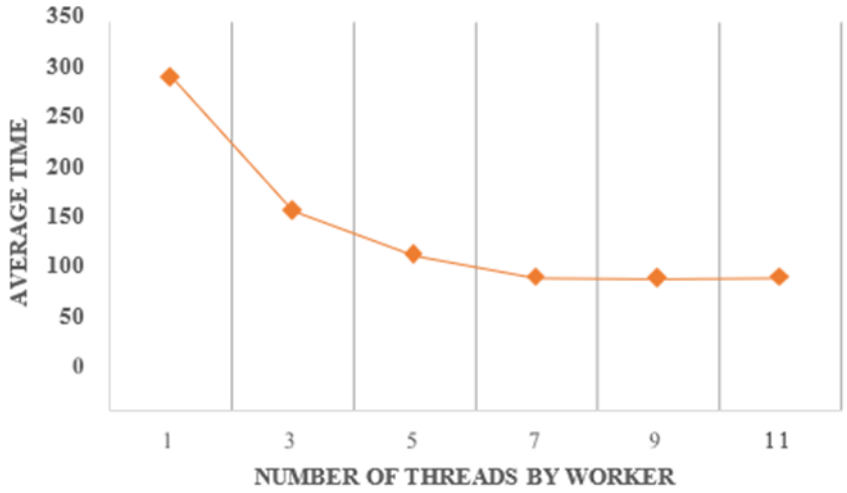

Figure 5: Average execution time of ETI job from a different source to JSON format.

5.2 Offline phase performance measurements

The proposed framework was evaluated using numerous nodes per cluster, as depicted in Figure 6 According to the figure, the greater the nodes per cluster, the slackened is loading time for each dataset. So, six nodes per cluster and one master node for that cluster were used in the evaluation and assessment process.

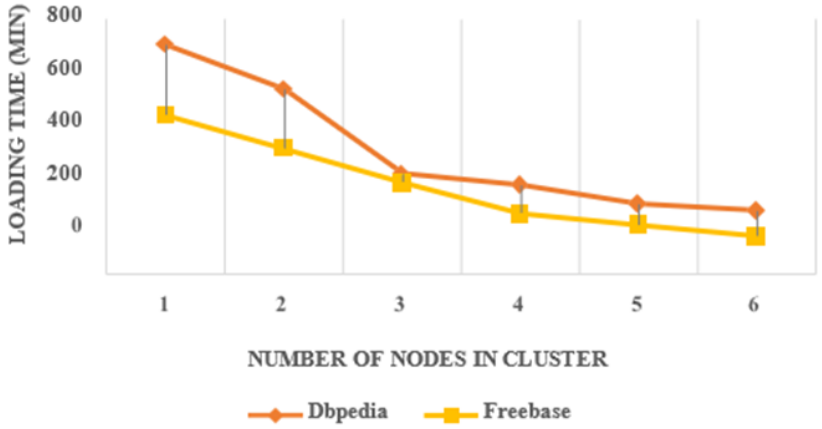

Figure 6: Loading time per cluster.

5.3 Online phase and query performance measurements

To scrutinize whether the proposed methods ameliorate the consequences and results, humans use different technologies to analyze humongous frequency queries in length. The effectuated four queries were cited under:

- **Query I**: In this test case, the system is triggered by a straightforward abstruse single word as query (e.g., network).
- **Query II**: In this query, the system is triggered by the same query as in query I (e.g., networks), but this query is expanded using the word net to extend the search area.
- **Query III**: In this query, the system is triggered by complex multiple words as a query (e.g., computer networks and protocols). Also, in this test case, all different proposed techniques are implemented.
- **Query IV**: In this query, the system is triggered by the same query as in query III (e.g., computer networks and protocols), but this query is expanded using the word net to extend the search area.

Table 2 provides the results of different online queries, and Table 3 details the compassion between the proposed system and other related systems.

Table 2: Result of different online queries.

Parameters	Query I	Query II	Query III	Query IV
Input Keywords	1	1	1	1
Extended Keywords	8	0	8	0
True Positive	8690	6481	8630	4991
False Positive	302	430	362	1120
Precision	0.966	0.937	0.959	0.889
Recall	0.938	0.943	0.921	0.957
F-score	0.952	0.935	0.94	0.921
Response Time at online phase (sec)	2.570	1.220	3.490	.400

Table 3: Compassion between the proposed system and other related systems.

Parameters	Swoogle	Falcon	Proposed
Support Ontology	✓	✓	✓
Semantic Relations	✓	✓	✓
Support Ranking	×	✓	✓
Handle Big Data	×	×	✓
High Precision	✓	×	✓
High Recall	×	✓	×
Response Time	Sluggish	Sluggish	Medium

6 Conclusion

As a matter of conjecture, this manuscript postulated a semantic retrieval framework for augmenting and enhancing knowledge and search area using the integration of more than one semantic source. The proposed framework is effectuated by deploying some modern technologies, including ontology graph, RDF, MAP-REDUCE technique implemented in Hadoop, Not only Structured Query Language (NoSQL) model using Hbase and proposed mathematical model for calculating semantic relations between subjects. Six slave nodes with one master node were utilized to scrutinize the framework. Two datasets (e.g., DBpedia and YAGO) were implemented to inculcate knowledge. Four queries were used in experiments that worked well to compute and test the proposed framework. Experimental results manifested a spectacular degree of precision and profound acuteness of cited tested queries in decent response time compared to related systems.

Declaration of Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding Declaration

This research did not receive any grants from governmental, private, or nonprofit funding bodies.

Author Contribution

Shishir Dua: Data curation, Writing–Original draft preparation, Methodology, Investigation, Software, Validation; **Kakoli Banerjee**: Conceptualization, Supervision, Validation, Writing- Reviewing and Editing

References

[1] C. Gavankar, T. Bhosale, D. Gunda, A. Chavan, and S. Hassan, "A Comparative Study of Semantic Search Systems," in *2020 International Conference on Computer Communication and Informatics (ICCCI)*, pp. 1–7, IEEE, jan 2020.

[2] Z. Pan, "Optimization of Information Retrieval Algorithm for Digital Library Based on Semantic Search Engine," in *2020 International Conference on Computer Engineering and Application (ICCEA)*, pp. 364–367, IEEE, mar 2020.

[3] A. Dramilio, C. Faustine, S. Sanjaya, and B. Soewito, "The Effect and Technique in Search Engine Optimization," in *2020 International Conference on Information Management and Technology (ICIMTech)*, pp. 348–353, IEEE, aug 2020.

[4] M. N. Asim, M. Wasim, M. U. Ghani Khan, N. Mahmood, and W. Mahmood, "The Use of Ontology in Retrieval: A Study on Textual, Multilingual, and Multimedia Retrieval," *IEEE Access*, vol. 7, pp. 21662–21686, 2019.

[5] A. Begdouri, O. Chergui, and D. Leclet-Groux, "A knowledge-based approach for keywords modeling into a semantic graph," *International Journal of Information Science & Technology, iJIST*, vol. 2, no. 1, pp. 2550–5114, 2018.

[6] M. M. El-Gayar, N. E. Mekky, A. Atwan, and H. Soliman, "Enhanced Search Engine Using Proposed Framework and Ranking Algorithm Based on Semantic Relations," *IEEE Access*, vol. 7, pp. 139337–139349, 2019.

[7] A. Nadeem, M. Hussain, and A. Iftikhar, "New Technique to Rank Without Off Page Search Engine Optimization," in *2020 IEEE 23rd International Multitopic Conference (INMIC)*, pp. 1–6, IEEE, nov 2020.

[8] M. Elgayar, "A Novel Knowledge-based Semantic Search Engine," *International journal of simulation: systems, science & technology*, vol. 20, pp. 9.1–9.9, oct 2019.

[9] Y. S. Negi and S. Kumar, "A Comparative Analysis of Keyword- and Semantic-Based Search Engines," in *Advances in Intelligent Systems and Computing*, vol. 243, pp. 727–736, 2014.

[10] B. R. Prasad and S. Agarwal, "Comparative Study of Big Data Computing and Storage Tools: A Review," *International Journal of Database Theory and Application*, vol. 9, pp. 45–66, jan 2016.

[11] D. Singh and C. K. Reddy, "A survey on platforms for big data analytics," *Journal of Big Data*, vol. 2, p. 8, dec 2015.

[12] Y. Qu and G. Cheng, "Falcons Concept Search: A Practical Search Engine for Web Ontologies," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 41, pp. 810–816, jul 2011.

[13] L. Ding, T. Finin, A. Joshi, R. Pan, R. S. Cost, Y. Peng, P. Reddivari, V. Doshi, and J. Sachs, "Swoogle Bibliographic Search," *Proceedings of the Thirteenth ACM conference on Information and knowledge management - CIKM '04*, p. 652, 2004.

[14] A. Hogan, A. Harth, J. Umbrich, S. Kinsella, A. Polleres, and S. Decker, "Searching and Browsing Linked Data with SWSE: The Semantic Web Search Engine," *SSRN Electronic Journal*, 2011.

[15] A. Fatima, C. Luca, and G. Wilson, "New Framework for Semantic Search Engine," in *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, pp. 446–451, IEEE, mar 2014.

[16] A. Oussous, F.-Z. Benjelloun, A. Ait Lahcen, and S. Belfkih, "Big Data technologies: A survey," *Journal of King Saud University - Computer and Information Sciences*, vol. 30, pp. 431–448, oct 2018.

[17] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop Distributed File System," in *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, pp. 1–10, IEEE, may 2010.

[18] D. Vohra, "Using Apache HBase," in *Pro Docker*, pp. 141–150, Berkeley, CA: Apress, 2016.