A Novel Knowledge-based Semantic Search Engine

M. M. El-Gayar [1], N. E. Mekky [1], A. Atwan[2], and H. Soliman[1]

[1] Faculty of Computer and Information Science, Mansoura University, Egypt
[2] Faculty of Computer Sciences and Information Technology, Northern Boarder University, KSA
Corresponding author email: (mostafa_elgayar@mans.edu.eg).

***Abstract* - The proliferation of websites and social media has increased the volume of big data of interconnected entities through numerous relationships. In this mesh scenario, millions of users need an engine to facilitate information searching and retrieval every day. So, analyzing, storing, and retrieving big data are considered the main challenging tasks. This paper recommends a knowledge-based semantic framework that merges some improved techniques such as preprocessing techniques, a proposed mathematical model of calculating semantic score and ontology-based semantic annotation. Furthermore, the wide column of Not Only SQL (NoSQL) indexing technique is used to overcome the enormous size of data. Several experiments were conducted to estimate the efficiency of the proposed model using two data sources: DBpedia and YAGO datasets. According to experimental outcomes, the proposed model attains high precision in reasonable time.**

Keywords - Big Data, Semantic Search Engine, Hadoop, HBase, MapReduce.

I. INTRODUCTION

Big data is a popular term where the data is big, from a verity of sources at a high rate. Various sources of big web data include websites, social media platforms, and wikis [1]. If any user needs to perform a query to search through these sources, then he/she may need to a search engine. Search engine moved through several ages. In the earliest age, the search engine relied on presenting information that accommodated with the input query only. Today, traditional search engines must analyze the text using machine learning techniques and then extract the valuable keywords based on the syntax form of words [2]. There is a significant variation between the syntax structure and semantic relation. The syntax structure is the study of grammar and how to say something. But, the semantic relation is the investigation of meaning behind the word. So, different words in the syntax structure may have the equivalent semantic meaning [3]. However, this domain is a hot topic because of modern challenges such as retrieving information not only by keywords but also by semantic relations [4] and handling big data in a reasonable time.

Useful tools for any knowledge-based search engine are Resource Description Framework (RDF) and Ontology Web Language (OWL) that presents the information fascinating and readable [5]–[7]. RDF scheme is a platform for representing information about references of data on the internet. Any RDF triple scheme consists of three elements. The subject element is the entity URI reference or a blank node. The object is the entity value. The predicate element is applied to indicate links between the subject and the object. There are a lot of current challenges and research points associated with search engines. Some focused points will be summarized as follows:

- Data is growing at a rapid rate. Capturing data is useful for analysis, but the extraction of useful information from big data is a challenging task.
- Scalability and availability of storing varied data is still a big challenge.
- Most traditional search engines focused on syntax similarity of keywords, not the semantic relation between keywords.

This paper has two significant contributions to overcome these challenges as follows:

i) The knowledge-based semantic search framework is suggested.
ii) A mathematical model for calculating the semantic score between different ontology subjects is proposed.

The rest of the paper is recognized in five parts. In section II, the background of some modern tools used for big data is introduced. Also, the background of some modern technologies is presented in section III. Furthermore, the previous works of the related semantic search engines are presented in section IV. In section V, the recommended knowledge-based semantic search framework and implemented algorithms are introduced. The environmental setup, the definition of different data sources, and performance evaluations are conducted in section VI. Finally, the conclusion is represented in section VII.

II. BACKGROUND OF MODERN TOOLS USED FOR BIG DATA

In this section, some of the modern tools that are implemented for solving the challenges of big data are introduced.

A. HADOOP

It is the open-source project of Apache software written in java. Hadoop uses Google's MapReduce as its foundation. Now Hadoop is a core part of many famous companies such as Yahoo, LinkedIn, Twitter, and Facebook [8] . Hadoop is used as a distributed master-slave architecture, as shown in figure 1. The tasks are initiated by master node and divided into small pieces among slave nodes within a cluster. Cluster is a collection of nodes (one master node and others are slaves). All slave nodes are connected with one master node. The following points are some important features of Hadoop [9]:

- High throughput is achieved because of batch processing.
- It achieves reliability because multiple copies of data are copied on different distributed machines.
- It is a fault tolerance because of the replication of data
- It does not work for real-time data processing.

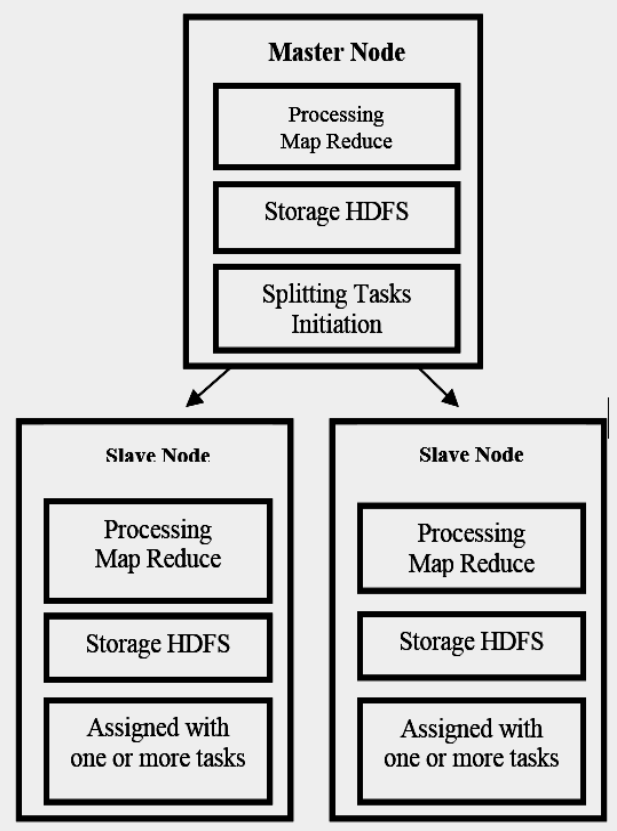

Fig.1. Hadoop Architecture

B. HBase

HBase [10] is also an open-source of Google's Big Table and supports CAP of non-relational distributed databases. It is used to provide horizontal scalability and enhance performance. HBase is a wide column NoSQL tool used for providing fault tolerance, scalability, and availability of data. Four main operations are allowed in HBase such as GET, PUT, SCAN, and DEL. GET operation used to retrieve single row and DEL is also used to delete one row. While PUT operation is applied to insert or update a single row.

C. DF

RDF is a framework that facilitates the integration of data from multiple and heterogeneous sources. It can also make data discoverable, easy to access, and reuse. The data model of RDF is a collection of triples. A triple of RDF consists of a subject, predict and object as represented in figure 2. Subject and object represent nodes in the ontology graph. Predict represents the relation between subject and object. Every source has its own RDF/XML file source and stored into the RDF repository [6].

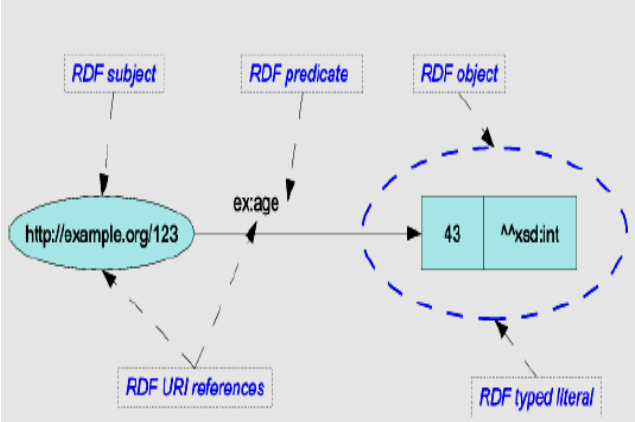

Fig.2. A triple exmple of RDF

III. BACKGROUND OF MODERN TECHNOLOGIES USED FOR BIG DATA

In this section, some of the modern technologies that are used to store and handle big data are introduced.

A. Distributed File System Architecture

In the Hadoop Distributed File System (HDFS) architecture [11], the client application interacts with the name node for metadata and initiates read and write activities on the file system. Then, name node communicates with the data node for applying read or write pipeline.

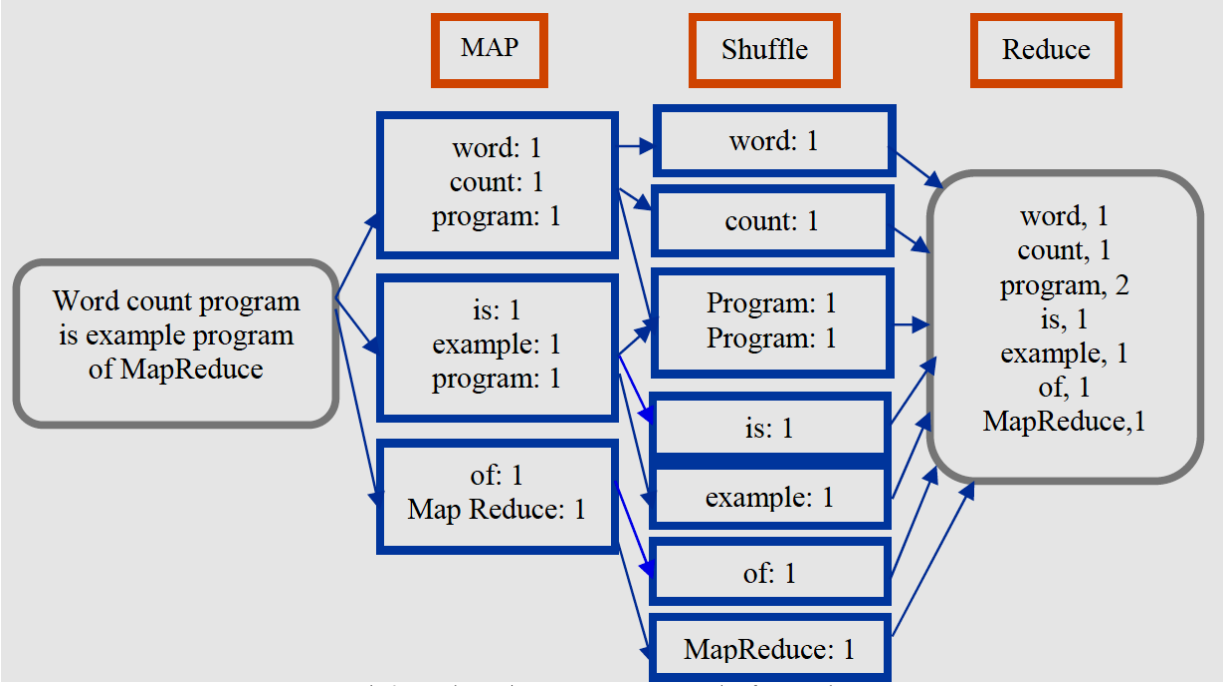

Fig.3. Word counting program as an example of MapReduce.

B. Batch Processing

Batch processing [12] is also known as offline indexing processing which splitting and processing huge data into batches. Online Analytical Processing (OLAP) is the famous tool that process data into batches.

C. MapReduce

MapReduce [13] is a programming model under Hadoop. Dean and Ghemawat proposed it at Google. It is used to process a massive amount of data in parallel form. Parallel data processing involves the simultaneous execution of many smaller tasks. The main goal is to reduce the execution time by dividing a single enormous task into many smaller tasks. It is divided into two common elements (Mappers and Reducers).

The mappers are used to generate typical combination pairs of key and value. The reducers are used to shuffle all intermediate values associated with the same key. Figure 3 shows a word count programming model as an example of MapReduce working model. For example, if we applied the MapReduce technique on a sample text "word count program is an example program of MapReduce" to find word counts. So, if we have three threads, then the sample text is divided into three lines. Each line has its mapper to count each word in the text. Then, shuffle as an intermediate function is used to combine the repeated words in all lines. Finally, reduce function is used to count final combined words to get the final result.

IV. RELATED WORKS

In this section, some of the old and new researches that explain how to develop different retrieval systems will be reviewed, while mentioning the advantages and disadvantages of each system.

Traditional search engines are divided into several types. The most popular kind of search engines is a keyword-based search engine such as (Google, Yahoo, and Bing). There is also another type of search engine called metasearch engine that performs the search and retrieval of data from other search engines such as Dogpile and Mamma. However, some problems and challenges are facing these types, including the inaccuracy of data with high recall. Furthermore, more information retrieval systems had some weaknesses, particularly concerning user experience, response time, and storage volume.

There is also another type of search engine that depends on the meaning of keywords and not just syntax. hakia is a search engine that relies on semantic ontology. it is considered a comprehensive search engine and works only for some specific purposes. one of its most important features is that it has achieved higher accuracy than conventional systems [14]. This engine is limited to research in specific areas such as medicine and law only and not all areas on the internet. So, this engine is considered a dictionary specialized in medical and legal meanings only.

The sematic search engine called falcons object search was able to achieve high accuracy in retrieving information based on semantic relationships between keywords and it was implemented by cheng et al [15]. this system is a search

engine based on slinking of keywords using rdf documents. this system contains the processes required to create a semantic search engine such as crawling, indexing, ranking results, retrieving data based on text analysis and extracting the appropriate summary for the user. the problem with this system is that it downloads the files, analyzes them and extracts the data during the user 's query, thus increasing the user' s response time. Furthermore, it cannot handle big data and doesn't support the semantic relation between related objects.

There is also a semantic search engine that became famous in 2004 called Swoogle [16]. It is divided into several major components, the most important of which is the development of a semantic crawler in addition to retrieval and analysis of data based on their semantic relationship. It allows metadata classification using rational surfer model. But, it cannot handle big data or different sources of data simultaneously compared with the proposed framework.

Web Search Engine (SWSE) [17] is another semantic model that also works to index and retrieve information based on word meanings and was developed by Hogan et al. It also contains the main components of a semantic search engine. But, SWSE has some weaknesses such as: (i) the system doesn't scale well, and (ii) the system doesn't robust in the face of heterogeneous, noisy, impudent, and possibly conflicting data collected from a large number of sources.

Laddha et al. [18] had suggested a tourism-based retrieval scheme based on getting the user's query and producing relevant outcomes for this query. This recommended scheme relies on its ontology and not depends solely on the keywords of the user's query. Fatima et al. [19] proposed a system that also depends on the semantic web that matches the data in a smart way based on the meanings of the words and not the degree of similarity only. This system depends on Page Rank algorithm to rank the results, which is a somewhat old technique and does not rank the results according to the meaning and its relation.

V. THE PROPOSED FRAMEWORK

A. Overview

The suggested framework is divided into four different stages, as shown in figure 4. The first stage contains various data sources, such as DBpedia and YAGO. The second stage involves preprocessing techniques such as extraction, transformation, and integration processes. Preprocessing is used to convert a variety of data sources into a single structure integrated source such as JSON.

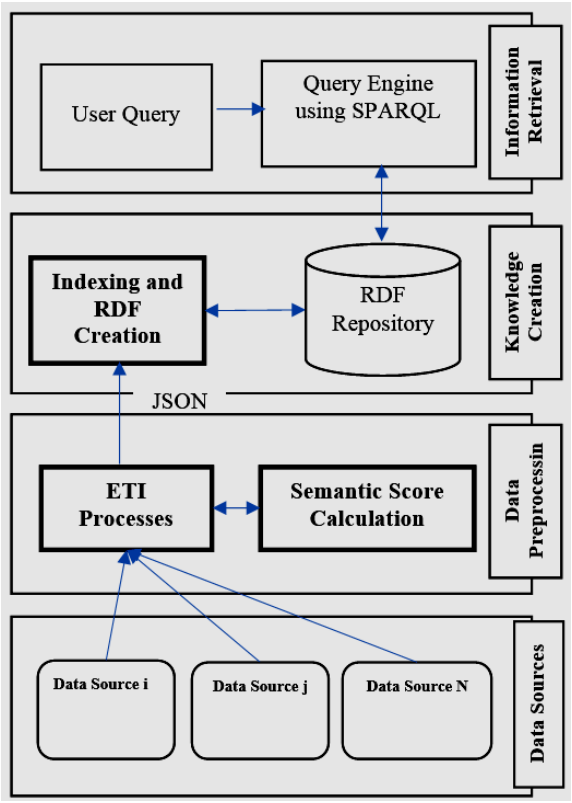

Fig.4. The Proposed Framework

Furthermore, the semantic score is calculated to assign weights of semantic relation between different subjects. Knowledge-based ontology graph is created in the third stage, and the JSON format is converted into RDF schema. Finally, when the user assigned query in a real-time, then the query engine use SPARQL Protocol and RDF Query Language (SPARQL) to retrieve data. As presented in figure 4, each bold box in this figure focused on one of the principal contributions of this paper. In the next subsections, each bold box is explained in details.

B. Data Preprocessing

The first bold box in figure 4 is the pre-processing block. It is consisted of two main processes such as extracting and transforming data from different datasets and converting it into uniform JSON format. The pre-processing job invokes on a master node to identify the schema of the different datasets. This step outputs a suit combined schema in JavaScript Object Notation (JSON) format. This part is one of the paper contributions.

Preprocessing step is launched on the master node and then disseminated on different slave nodes using the MapReduce parallel processing method. The extract function includes obtaining data from the source dataset, which should be exact and correct. The transform function produces a sequence of complex data cleaning and reformation it into JSON format.

The Proposed algorithm 1 explains how data can be selected and extracted from the dataset. In the extraction function, the input parameters are the dataset and the number of threads allocated. The production of this function is extracted data according to batch size.

Algorithm#1 Extract Function
Input ← Dataset (DS_i), Number of Threads (NT)
Output ← Extracted Data (ED)

```
1.  Start Extract function
2.    DB ← DSi .open
3.    Total ←DB. collection. Names. Count ()
4.    L ← Total / NT
5.    For i ← 0 to Total do
6.      initial ← L * i
7.      Data ← Read data (batch size)
8.      ED ← Extract (process id, Data,
                      DSi , initial, L, schema)
9.    End for
10.   Return ED
11. End Extract function
```

Transform function can list all the variations in the dataset and organize the schema into a JSON form, as given in proposed algorithm 2. In the transformation function, the input parameters are the dataset, the number of threads allocated from master node to slave nodes and obtained data from the previous step. The production of this function is blended data in JSON format.

C. Semantic Score Calculation

The weight score determination based on semantic relation is one of the principal contributions of this article. Semantic similarity score (triple S) in equation (1) is applied to estimate the similarity weight score between two nodes or objects (I and J). The semantic similarity score is a relationship score based on the semantic relation of links entered of each node or object in the ontology graph. It has a fraction value between zero and one.

Algorithm#2 Transform Function
Input ← Dataset (DS_i), Extracted Data, Number of Threads (NT)
Output ← JSON

```
1.  Start Transform function
2.    Data ← Extracted Data. Length
3.    L ← Data / PC
4.    j ← 1
5.    While j < L do
6.      Stemming (Extracted Data)
7.      Remove Duplicates (Extracted Data)
8.      Initialize Schema
9.      j ++
10.   End While
11.   JSON ← Bind Schema
12.   Return JSON
13. End Transform function
```

$$SSS(Ni, Nj) = 1 - \left(\frac{\max(\log Ni, \log Nj) - \log(Ni \cap Nj)}{\log T - \min(\log Ni, \log Nj)}\right) \quad (1)$$

Where:
- Ni is the number of input links of node I.
- Nj is the number of input links of node J.
- T is the total number of objects in the ontology graph.

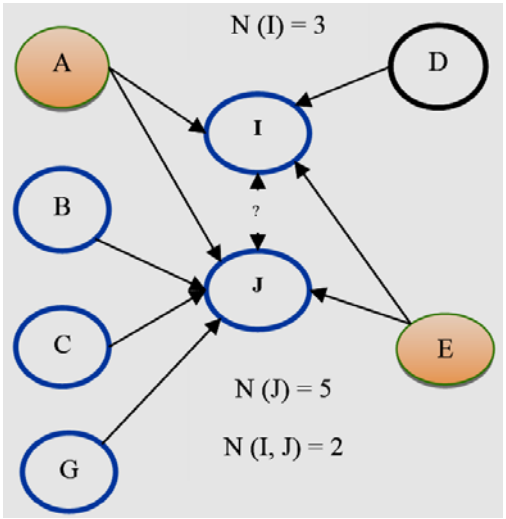

Fig.5. An example of calculating semantic similarity score between two nodes using six nodes and two intersection nodes

Figure 5 shows an illustration of determining the semantic weighting score between two nodes (I and J). If the node (I) has three incoming relations from different nodes (A, D, and E). So, the semantic relationship of the node I was three.

Similarly, if the node J has five incoming relations from different nodes, such as (A, B, C, E, and G). So, the semantic relationship of the node J was five. However, there are two nodes (A and F) related to both source nodes (I, J). So, the semantic relationship of both nodes (I and J) was two. If the graph has one hundred nodes, then the Semantic Similarity Score (triple S) by using equation (1) equal to 0.81 which means high similarity.

$$SSS(i,j) = 1 - \left(\frac{\max(\log 3, \log 5) - \log(2)}{\log 100 - \min(\log 3, \log 5)}\right)$$

$$= 1 - \left(\frac{\log 5 - \log 2}{\log 100 - \log 3}\right) = 0.81$$

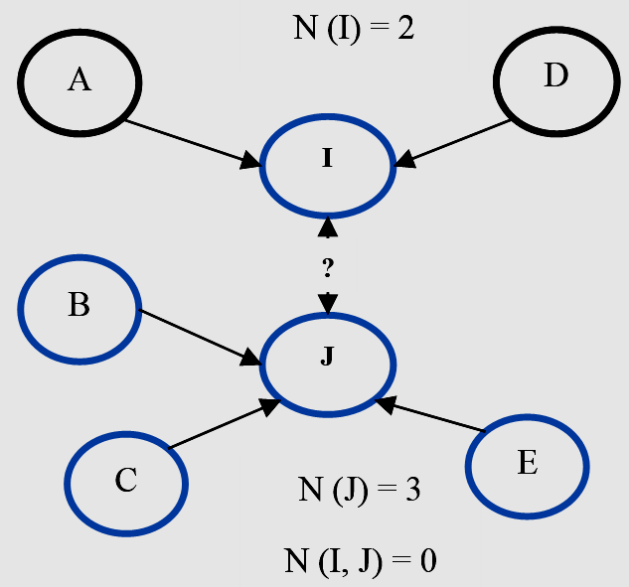

Fig.6. An example of calculating semantic similarity score between two nodes using five nodes and zero intersection nodes

Figure 6 also shows another example of determining the semantic weighting score between two nodes (I and J). If the node (I) has two incoming relations from different nodes (A and D). So, the semantic relationship of the node (I) was two. Similarly, if the node (J) has three incoming relations from different nodes, such as (B, C, and E). So, the semantic relationship of the node (J) was three. However, there are zero nodes related to both source nodes (I, J). So, the semantic relationship of both nodes (I and J) was zero and log zero is infinity. If the graph has one hundred nodes, then the Semantic Similarity Score (triple S) by using equation 1 equal to zero which means no similarity and no relationship between these two nodes.

$$SSS(i,j) = 1 - \left(\frac{\max(\log 2, \log 3) - \log(0)}{\log 100 - \min(\log 4, \log 5)}\right)$$

$$= 1 - \left(\frac{\log 3 - \log 0}{\log 100 - \log 2}\right) = zero$$

D. Indexing Part

The ontology structure is divided into an essential pair of actions. The initial action is the origin node creation method. In this method, the graph is initially in the style of a tree form. The root or origin node is the heart of the domain, and then the related nodes are joined in the given hierarchy. The next action is the production method of objects as children nodes.

In the indexing part, we used HBase to store data. The translation from the logical data model to the physical One is represented in figure 7. Each column family within a region translates into separate HDFS directories called Stores. Each Store is made up of multiple on-disk StoreFiles. Each entry in these files is a key-value pair in which the key is composed of <logical rowKey,column family > and the value contains the associated logical cell value.

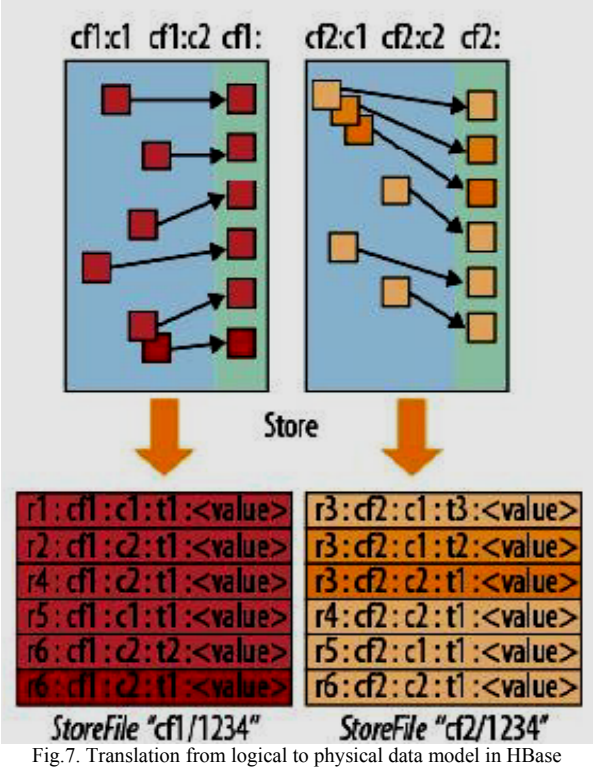

Fig.7. Translation from logical to physical data model in HBase

VI. EXPERIMENTAL RESULTS

A. Setup

The recommended framework is assessed on a cluster of seven machines. The cluster has included one master machine, and others were the slave machines. The master device has an Intel Core i-7 processor using 8GB RAM. The slave devices had an Intel Core i-5 processor with 4GB RAM for each. Hadoop version 2.9.2 was applied, and it was run with three operators per MAP operation and two operators per REDUCE process. Also, Apache HBase version 2.2.0 was applied. Figure 8 presents the average preprocessing execution time for several operators. According to figure 8, six operators were applied because they realized the minimum loading time.

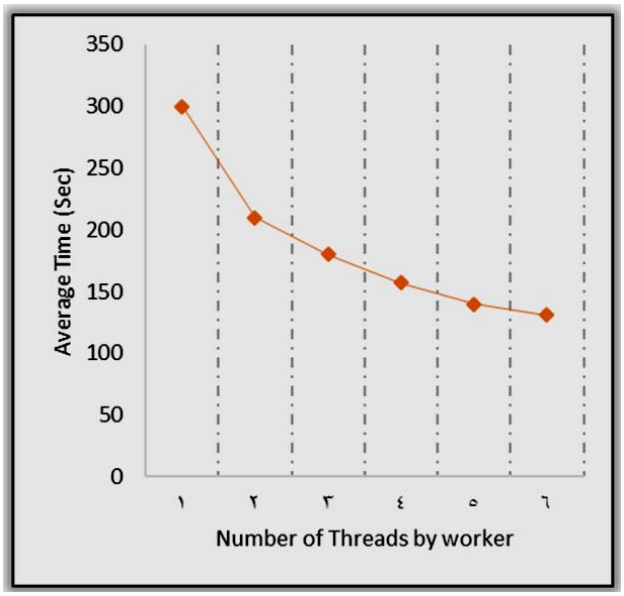

Fig.8. Average time of preprocessing task using different number of threads

B. Datasets

The proposed framework was evaluated using two real world datasets (DBpedia version 2016-04 and YAGO 2015-03). Figure 9 shows the number of entities per domain for each dataset [20].

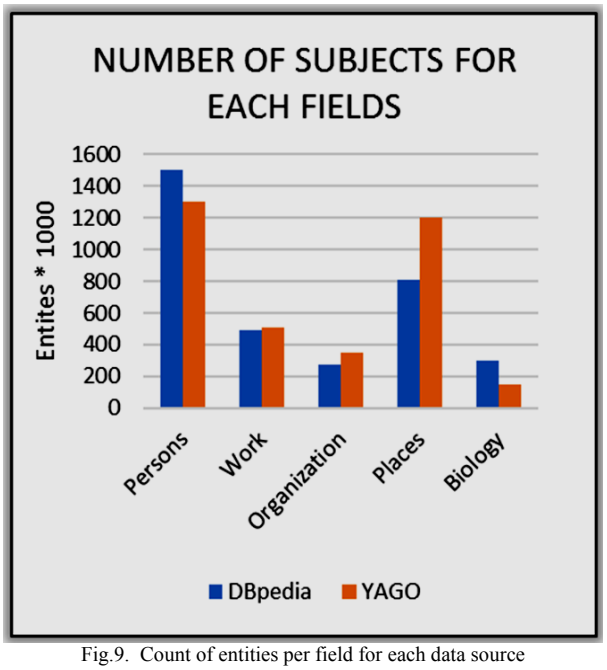

Fig.9. Count of entities per field for each data source

C. System Evaluation

The total throughput and loading time by using six nodes for each dataset's quad samples is shown in table I. The main metric used in this evaluation step is arithmetic mean. Overall, we notice that Pre-processing and indexing processes scale well using Map-Reduce technique. Figure 10 shows the throughput of entities per domain for each dataset. Figure 11 shows the loading time of entities per domain for each dataset.

TABLE I. THROUGHPUT AND LOADING TIME FOR EACH SAMPLE FROM EACH DATASET USING 6 NODES IN CLUSTER

Dataset-Sample Size	Throughput Using MAP-Reduce (Quads/sec)	Loading Time (Min.)
DBpedia - using 10 million entities	11500	29.480
DBpedia - using 50 million entities	25804	105.055
YAGO - using 10 million entities	13890	26.032
YAGO - using 50 million entities	26213	95.942

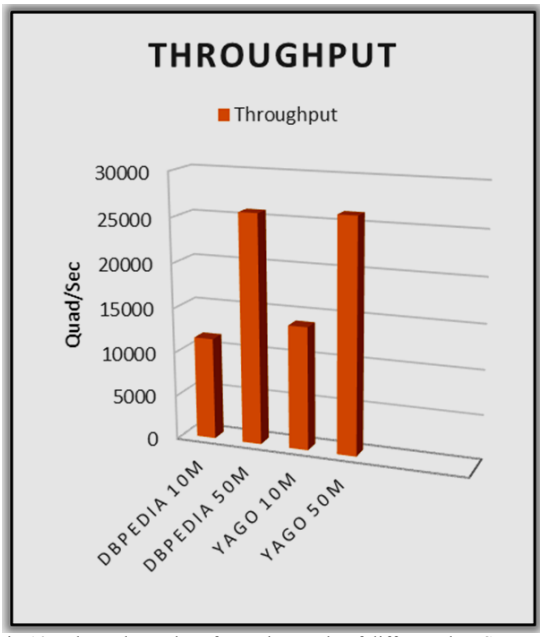

Fig.10. Throughput chart for each sample of different data Sources

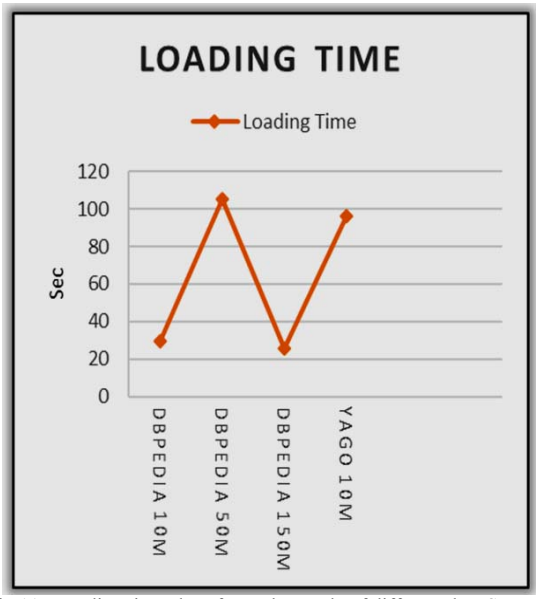

Fig.11. Loading time chart for each sample of different data Sources

D. Implemented Queries

To see if the recommended system enhances the results, we should test several queries. Two different implemented queries are conducted as follows:

1) Query 1: In this query, the query engine is issued by a single unclear word as query (e.g., engine).

2) Query 2: In this query, the query engine is issued by multiple words as query (e.g. semantic search engine).

E. Performance Measures

In this sub-section, the performance results of applying the previously mentioned two test cases are presented in table II and figure 12. The performance results were measured according to the following equations (1), (2) and (3):

$$precision = \frac{TP}{TP+FP} \quad (2)$$

$$recall = \frac{TP}{TP+FN} \quad (3)$$

$$F-score = \frac{2PR}{P+R} \quad (4)$$

Where, TP, FP, TN, and FN which refer to true positive, false positive, true negative and false negative respectively.

TABLE II. PREFORMANCE TABLE OF DIFFERENT IMPLEMENTED TEST CASES

	Query 1	**Query 2**
Precision	0.92	0.95
Recall	0.94	0.93
F-score	0.93	0.94
Response Time (sec)	3.52	2.45

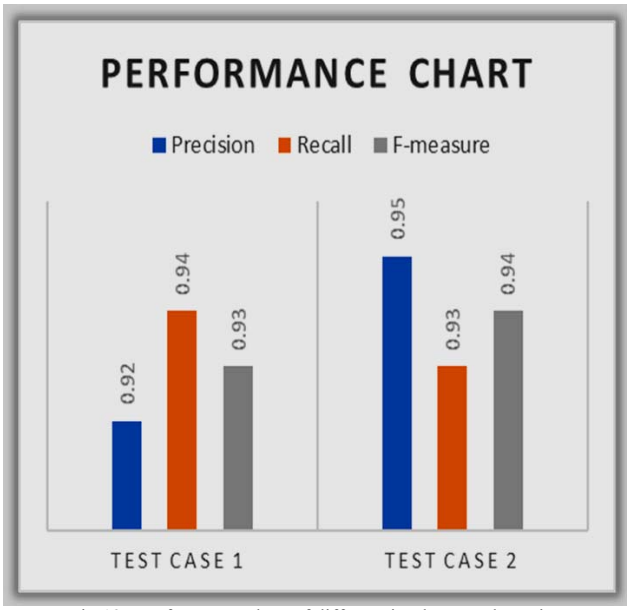

Fig.12. Preformance chart of different implemented queries

F. Comparision Between Proposed System and other Related Systems

In this sub-section, the comparison among the proposed system and other associated systems is conducted. Every system has its advantages and disadvantages in different criteria, as described in table III.

TABLE III. COMPARISON BETWEEN PROPOSED SYSTEM AND OTHER RELATED SEMANTIC SYSTEMS USING DIFFERENT CRITERIA.

	Swoogle System	**Falcon System**	**Proposed System**
Handling Big Data	No	No	Yes
Supporting Ontology Graph	Yes	Yes	Yes
Calculating Semantic Weights	Yes	No	Yes
Achieving High Precision	Yes	No	Yes
Achieving High Recall	No	Yes	No

VII. CONCLUSION

In this paper we introduced a framework for enhancing knowledge-based on semantics. The recommended framework is performed using some modern techniques such as MapReduce procedure executed in Hadoop and NoSQL wide column form using Hbase. Furthermore, a novel of mathematical equation for measuring semantic weights between different nodes is introduced. Two datasets (e.g., DBpedia and YAGO) are used to build knowledge. According to system evaluation, the proposed system achieved high throughput.

REFERENCES

[1] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *J. Big Data*, vol. 2, no. 1, Dec. 2015.

[2] Y. S. Negi and S. Kumar, "A comparative analysis of keywordand semantic-based search engines," in *Advances in Intelligent Systems and Computing*, 2014, vol. 243, pp. 727–736.

[3] J. M. Kassim and M. Rahmany, "Introduction to semantic search engine," in *Proceedings of the 2009 International Conference on Electrical Engineering and Informatics, ICEEI 2009*, 2009, vol. 2, pp. 380–386.

[4] A. Begdouri, O. Chergui, and D. Leclet-Groux, "A knowledge-based approach for keywords modeling into a semantic graph," 2018.

[5] "OWL," in *Semantic Web: Concepts, Technologies and Applications*, London: Springer London, pp. 81–103.

[6] "RDF and RDF Schema," in *Semantic Web: Concepts, Technologies and Applications*, London: Springer London, pp. 57–79.

[7] M. N. Asim, M. Wasim, M. U. G. 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.

[8] B. R. Prasad and S. Agarwal, "Comparative Study of Big Data Computing and Storage Tools: A Review," *Int. J. Database Theory Appl.*, vol. 9, no. 1, pp. 45–66, Jan. 2016.

[9] 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, no. 4. King Saud bin Abdulaziz University, pp. 431–448, 01-Oct-2018.

[10] D. Vohra and D. Vohra, "Using Apache HBase," in *Pro Docker*, Apress, 2016, pp. 141–150.

[11] 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, MSST2010*, 2010.

[12] S. Shahrivari, "Beyond batch processing: Towards real-time and streaming big data," *Computers*, vol. 3, no. 4. MDPI AG, pp. 117–129, 01-Dec-2014.

[13] D. Singh and C. K. Reddy, "A survey on platforms for big data analytics," *J. Big Data*, vol. 2, no. 1, Dec. 2015.

[14] V. Bevilacqua, V. Santarcangelo, A. Magarelli, A. Bianco, G. Mastronardi, and E. Cascini, "A semantic search framework for document retrievals (literature, art and history) based on Thesaurus multiwordnet like," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011, vol. 6838 LNCS, pp. 456–463.

[15] Y. Qu and G. Cheng, "Falcons concept search: A practical search engine for web ontologies," *IEEE Trans. Syst. Man, Cybern. Part ASystems Humans*, vol. 41, no. 4, pp. 810–816, Jul. 2011.

[16] L. Ding *et al.*, "Swoogle Bibliographic Search," *Proceedings of the Thirteenth ACM conference on Information and knowledge management - CIKM '04*. ACM Press, New York, New York, USA, p. 652, 2004.

[17] 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," *J. Web Semant.*, vol. 9, no. 4, pp. 365–401, Dec. 2011.

[18] S. S. Laddha and P. M. Jawandhiya, "Semantic tourism information retrieval interface," in *2017 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2017*, 2017, vol. 2017–January, pp. 694–697.

[19] A. Fatima, C. Luca, and G. Wilson, "New framework for semantic search engine," in *Proceedings - UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, UKSim 2014*, 2014, pp. 446–451.

[20] M. Färber, F. Bartscherer, C. Menne, and A. Rettinger, "Linked data quality of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO," *Semant. Web*, vol. 9, no. 1, pp. 77–129, 2018.