IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 21, 2019, accepted September 12, 2019, date of publication September 17, 2019, date of current version October 7, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2941937

Enhanced Search Engine Using Proposed Framework and Ranking Algorithm Based on Semantic Relations

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, Mansoura 35516, Egypt
[2]Faculty of Computer Sciences and Information Technology, Northern Border University, Arar 73222, Saudi Arabia

Corresponding author: M. M. El-Gayar (mostafa_elgayar@mans.edu.eg)

ABSTRACT Today, most users need search engines to facilitate search and information retrieval processes. Unfortunately, traditional search engines have a significant challenge that they should retrieve high-precision results for a specific unclear query at a minimum response time. Also, a traditional search engine cannot expand a small, ambiguous query based on the meaning of each keyword and their semantic relationship. Therefore, this paper proposes a comprehensive search engine framework that combines the benefits of both a keyword-based and a semantic ontology-based search engine. The main contributions of this work are developing an algorithm for ranking results based on fuzzy membership value and a mathematical model of exploring a semantic relationship between different keywords. In the conducting experiments, eight different test cases were implemented to evaluate the proposed system. Executed test cases have achieved a precision rate of 97% with appropriate response time compared to the relevant systems.

INDEX TERMS Information retrieval, semantic search, semantic ranker, search engine, resource description framework.

I. INTRODUCTION

The proliferation of the Internet and social media have added new challenges to the traditional search engines that must be addressed. Information retrieval system may be affected by the user input query, and the retrieved topics may be different and not related to the meaning of the search subject [1]. Also, the retrieved results may be affected by the ranking process [2] based on their relevance and semantic relation to the subject of the search [3]–[5]. So, there are several significant challenges in this field that can be summarized as follows [6]–[8]:

i) Most current search engines rely on indexing and retrieving different pages on keywords only that are often small, unclear, and does not reflect the meaning of the topic.
ii) Different search engines also rely on the user's query itself to search in the databases without increasing the keywords in this query to expand the scope of the search and improve the accuracy of the results.
iii) The ranking procedure is a significant issue and not a concern for some search engines. If a page is relevant to a query but is ranked very low (e.g., below top 20), then the user is unlikely to look at the page.

The idea of semantic search aims to solve the limitations and problems associated with traditional search based on keywords. Semantic search relies on the purpose of retrieving information using tags with adding the advantage of linking these tags and know the meaning behind and add more tags that improve the search results [9]. An ontology-based approach used to represent the vocabularies and relationships between semantic entities. Ontology describes the elements that exist in any field or area to represent semantic relation [10]–[14]. The contributions in this paper are divided into four main categories and can be summarized as follows:

i) Semantic web retrieval framework is proposed which improves the input unclear query and retrieves the relevant data with high precision in a fair time using the techniques such as MapReduce, Latent Dirichlet Allocation (LDA), Resource Description Framework (RDF) and Not Only SQL (NoSQL) document-oriented Model.

The associate editor coordinating the review of this manuscript and approving it for publication was Wajahat Ali Khan.

ii) A novel of preprocessing algorithm is developed to extract useful keywords from crawled pages.
iii) A novel of ranking algorithm and mathematical model of calculating semantic score is developed to order and classify the relevant results of unclear query based on semantic relations.
iv) Enhancing the query engine using the ontology analyzer and Wordnet to increase the input keywords.

The remaining of this article is divided as follows: Section II reviews previous work of traditional, semantic information retrieval schemes and focused on ranking methods. Section III describes the global challenges that meet search systems. Section IV illustrated the various elements of the suggested framework and suggested ranking method. Section V describes test cases and experiment outcomes, and evaluation will be performed. Finally, the last section will provide conclusions and references.

II. PREVIOUS WORK

In this section, some of the researches that are interested in information retrieval with the standard and modern ways will be reviewed. After that, some systems related to semantic retrieval will be discussed and focused on its weaknesses. Finally, we will focus on research that is related to the semantic ranking of results.

Most of the current search mechanisms are classified into several similar items in their components such as crawling and indexing but vary in the way they work. Search systems are organized into categories such as search engines such as Google, and directories such as Yahoo, and Meta Search systems. Most of the standard search mechanisms are very popular, but their results are sometimes inaccurate which have a lower precision and high recall. They lack to find the meanings of terms and expressions used in web pages and their relationships. The problem lies in the existence of words that have many meanings in natural languages [8], [15], [16].

Modern and intelligent search systems, like Swoogle, SWSE, Falcons Object Search, . . . etc., are designed based on the semantic approach to overcome the traditional problems. Swoogle is a system that relies on semantic crawling and indexing of web documents. It is divided into four main components: data discovery, the creation of metadata, data analysis and retrieval [17]. The most significant drawback of this system is that it is not a general-purpose search system and is limited to predefined ontologies files [15]. Also, it has some weaknesses such as weak indexing of massively large data and time-consuming of query response as discussed in section 5.

Hogan *et al.* [18] have designed a model of the semantic search system called Semantic Web Search Engine (SWSE). It is a comprehensive search system that provides services similar to traditional search engines based on RDF and link related data. It consists of the crawling process, indexing, Reasoning, and retrieval phase. But, SWSE has some weaknesses such as weak ranking of records because the ranking step is proceeding before the indexing step.

Qu and Cheng [19] have built a retrieval model based on semantic relation called Falcons Object Search. For every detected object, the system produces an extensive implicit record containing the textual summaries obtained from its RDF representation. Unfortunately, this model is not carried by a ranking process to rate these objects that related to the submitted query.

Hakia [15] is another system that acts as a comprehensive semantic engine that works for general purposes. It is called a search engine based on meaning rather than search terms [15]. It consists of many components like query processor, ontology analyzer, QDex storage, and ranking approach. Unfortunately, this model has a scalability and indexing weakness because of virtual contents and dynamic contents.

A semantic search engine called Semantic Illegal Content Hunter (SICH) has been developed to detect illegal content and with the financial support of the European Crime Prevention and Control Organization [20]. The specific purpose of this engine is to analyze the semantic text and identify the illegal content. This engine consists of three stages, and the first stage includes the collecting and indexing of information. The second phase is the data analysis and is not sufficient in massive large data. Finally, a third phase that interacts with the user to retrieve the data but is not a way to rank outcomes [20].

Al-Yahya *et al.* [21] had created a model based on the ontology of Arabic dictionaries to search the texts in the Holy Quran. This proposed model is based on semantics, but in the field of Arabic, especially in the Quran, and although it achieves accurate results, it does not rank or index the results, but only depends on the similarity match.

Al-Safadi *et al.* [22] had proposed a systematic system for an Arab search engine based on the field of ontology, but only within Arab blogs. It categorized the Arabic language into a series of classes, characteristics, and relationships. But it works on Arabic only and in specific areas as it takes a lot of time to enforce the task.

Medhat *et al.* [23] had developed a proposed an Arabic semantic search system that relies on four main components, such as data acquisition, an indexer, classifier and retrieval engine. But it depends on acquired data from structured governmental data, which is not large, and not facing many of the problems in any comprehensive search engine.

Laddha and Jawandhiya [24] had proposed a tourism-based search system based on understanding the user's query and providing relative results for this query. However, this system is different from our proposed method. It is a specific system in the field of tourism only which it is not a comprehensive model and did not rank the results or classified the topics to facilitate the indexing and retrieval of large data.

Abatal *et al.* [25] had proposed an intelligent system based on semantic research and the integration between the ontology and cloud computing of health services. In this research, authors have been able to publish and share medical reports quickly and accurately. But despite the excellent work done by the researchers, it lacks some important points, such as

dealing with large data, splitting reports before indexing, and the lack of system to arrange results in a way that makes it easier for the user to get information quickly and accurately.

Both Page *et al.* [26] and Duhan *et al.* [27] developed a method for ranking pages indexed through the Internet. Google works in this way, and the founders of Google search engine are the two developers of this algorithm. This algorithm state that if a page that contains important links and topics then its links to other pages become important also.

But this method has become impractical at present and because of the fabrication pages with fabricated links and not related to the query subject. In [28] authors proposed a weighted page rank mechanism based on visited links.

In [29], authors were focused on the study area that intended to avoid confusing results by ranking query outcomes of a sparkle Protocol and RDF Query Language (SPARQL) query. Inappropriate, they produced a novel method that supports both keyword exploration and extending ranking measures.

Finally, authors [30] suggested a novel procedure of ranking process. The characteristics needed for ranking the relevance of entities are the number of subjects, the number of objects, average frequencies and number of literals. However, this approach is not fitting for semantic relations.

III. MAIN CHALLENGES IN SEMANTIC SEARCH RETRIEVAL SYSTEM

As mentioned previously, most of the traditional or modern search engines face many problems. Some of these challenges or issues will be addressed in this section and will be solved using the proposed framework in the next section.

A. EFFICIENCY

Search system's accuracy depends upon the volume of documents to resemble, related outcomes, matching records, and response time. The main benefit of smart semantic search systems is retrieving the most significant results with high precision and lower recall. The returned results from Google had a higher recall than precision rate [8], [15], [22].

B. RANKING PROCEDUR

The focal concept of the semantic web retrieval system is to recover the various important (most exactness) and actual outcomes as a reply to a query. A ranking approach is a challenging task given that there are ''more than 12.3 billion sources on the internet'' at the moment of typing [8], [27], and an appropriate query on a system may declare thousands of outcomes. It is crucial to classify and rate the recovered documents effectively [2].

C. EXPANSIBILIT

Scalability or expansibility is the ability of a system to manipulate a quickly expanding the volume of data. Relational databases are structured but not scale well. But, expansibility for records in a semantic web offers extra difficulties because of the massive volume of unstructured related data [8], [31], [32].

D. UNBALANCED BIG DAT

One of the challenges of search engines is unbalanced big data (one of the most challenge of big data). The big challenge that many researchers have been working on lately is dealing with large unbalanced and unregulated data. Search result from any search directory or search engine may lead to many randomized results which are not well categorized [32], [33].

IV. PROPOSED FRAMEWORK AND ALGORITHM

The recommended framework is introduced in a modular way and reasonably formed of two distinct stages. Firstly, the backend phase is comprising crawling, indexing, and ranking stages where the servers are operated separately from users. Secondly, the frontend or retrieval phase is containing browsing and retrieving stages where a user can operate directly with the backend servers. As shown in Fig.1, Each blue box of this framework focuses on one contribution of this paper. The next subsections describe each part of the proposed framework in details.

A. CRAWLING PART IN BACKEND PHASE

Backend phase is divided into four main blocks, which are crawling, data mining, indexing, and ranking processes. These blocks have been performed away from users (not in a real-time) because they consume a long time.

The first critical process in that phase is the crawling process. In this process, we use MapReduce [35] (a tool used to handle a large amount of data) as a multithreaded programming model to support parallelization and divide and conquer style over the input data (page file).

The first sub-step of crawling procedure is extracted useful tags as defined in Algorithm 1. The input parameter is the entry page, and the output parameters are some collection of arrays. The primary step of this algorithm is focused on traverses TREE objects (Nodes) of entry page (index or threaded page) to an array of objects and data nodes using DOM-TREE. After that, the looping process (through an array of DOM objects) is used to filter unused tags with depth less than 3 (such as HTML, BODY, HEADER, ... etc.). Useful data nodes that pass-through filtration process are stored in an array of extracted nodes. But, if the tag is a link tag, then it is used to recrawling as a new thread. The pre-processing step is discussed in the next paragraph.

The second sub-steps of crawling procedure are parsing and cleaning steps (called pre-processing steps). The pre-processing procedure is used to parse data regions from useful DOM objects (nodes), cleaning data records and obtain valuable keywords as defined in Algorithm 2. This module is a blue box as in Fig.1. We can simplify parsing and cleaning processing method into three main processes. Firstly, the map function divides the input, for example, a DIV or TITLE object containing many texts, into chunks of independent data.

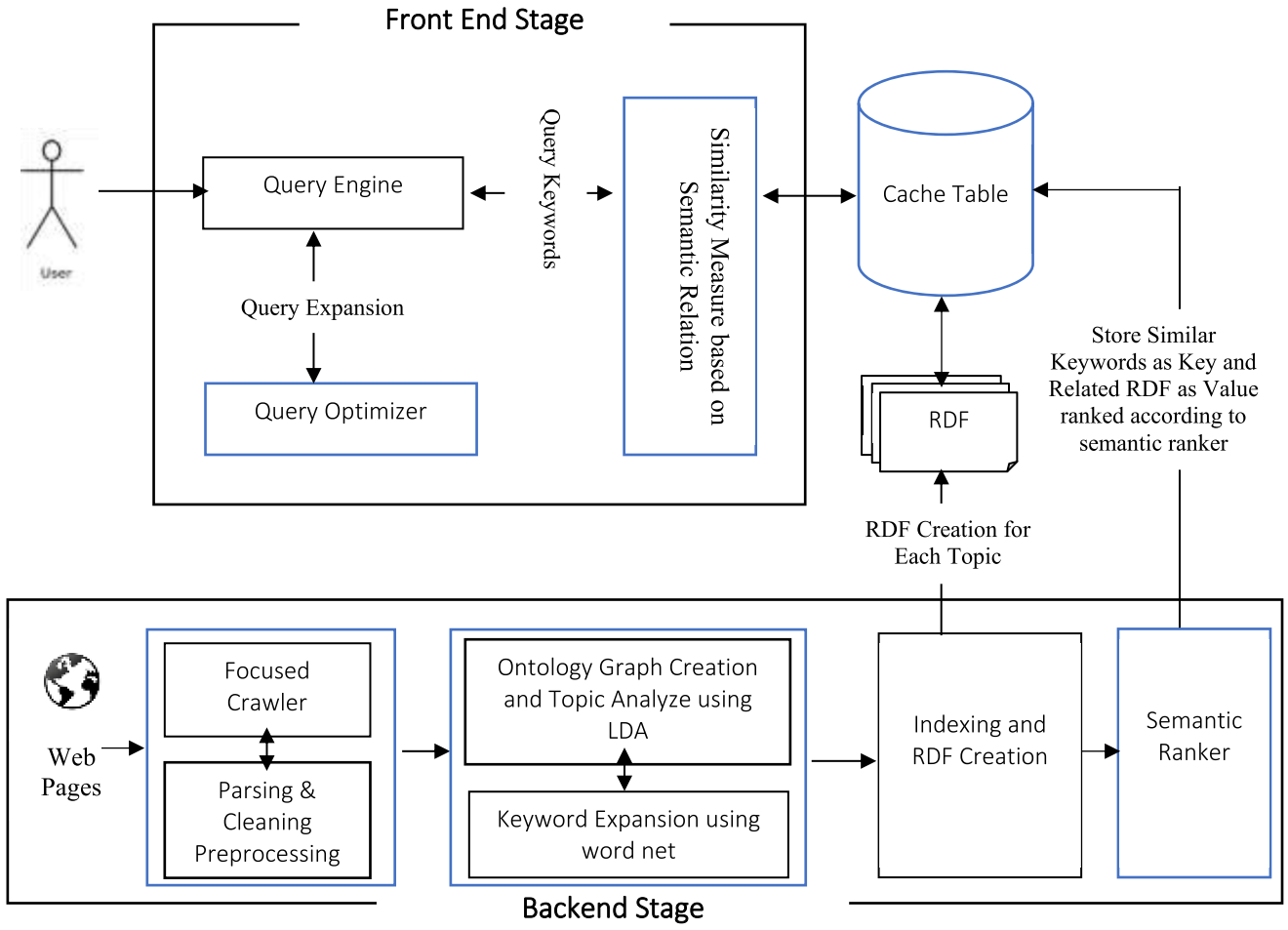

FIGURE 1. Logical architecture diagram of proposed framework.

For each separate section of this data, a pair of key and value is calculated. Secondly, these pairs send the key and value to parallel maps whose task is to search for an event, word, or several synonyms. The Mapper outputs one or more intermediate key-value pairs. Thirdly, the reduce function is used to process the intermediate output data (i.e., filtering, summarizing, sorting, caching, taking an average, or finding the maximum). In this part as mentioned before, crawling procedure with all sub-steps are implemented using Hadoop and MAP-Reduce tools to cover the time consumption (Efficiency challenge) that noted in section 3.

The first input parameter of pre-processing algorithm (named Algorithm 2) is node which extracted from the previous crawling step. The second input parameter is M. M is the maximum number of specialized nodes. τ is the similarity threshold. The object comparison can be done using string distance. When the data regions are defined, then the stop words removal and stemming processes (called cleaning methods) are performed. Stemming is the process of removing inflection of words so that we have to convert different words in the same root or stem word which contains the similar meaning.

Fig.2 illustration an example of DOM TREE specialized nodes and data regions. Specialized nodes are nodes that have the same parent and they are adjacent in the DOM tree. Data region is a collection of specialized nodes or objects with following properties:

1) The specialized nodes or objects all have the same parent node or object.
2) The specialized nodes or objects are all adjacent.
3) Similarity between adjacent specialized nodes or objects is greater than the threshold $\boldsymbol{\tau}$

For example, in fig.2, node 1, node 2, node 3 and node 4 are represented BODY, TABLE, IMG and DIV as parent nodes respectively. Node 5, node 6 and node 7 are represented TR tags as child nodes of TABLE nodes and considered as first data regions. Also, Node 8 and node 9 are represented as P (paragraph) tags and considered as second data regions. So, all gray nodes in fig.2 are considered as specialized node with different data regions.

B. TOPIC ANALYZER PART (BACKEND PHASE

Topic Analyzer comes after the crawling which considered as one of our main contributions. Association rule method

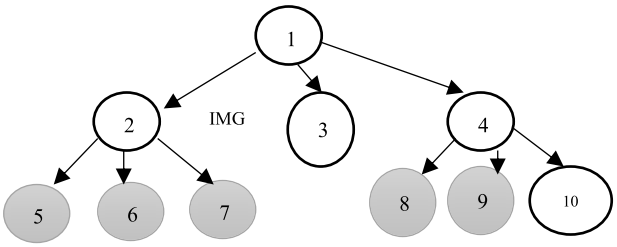

FIGURE 2. An example of specialized nodes and data regions.

Algorithm 1 Crawling Algorithm

Input : ← URL (entry page)
Output: ← DOA (DOM Objects Array from Threaded Pages) and ENA (Extracted Nodes Array)

```
DOA = φ;
ENA = φ;
Type = URL Detection (URL);
if Type == index page OR threaded Page then
    New Thread ≪ URL;
    DOA = DOM Objects of entry page;
    for i: DOA. Length do
        if Tree Depth (DOA[i]) >= 3 then
            if DOA[i] == Link Tag then
                Crawling (DOA[i]) // recursion
            else
                ENA[i] = DOA[i]
            end
        end
    end
end
Return TPA, ETA and Rx
```

Algorithm 2 Pre-Processing Algorithm

Input : ← Objects (Array of Crawled Objects), M, τ
Output: ← DR (Data Regions), KA (Keywords Array)

```
for i ← Objects.length do
    Match (Object[i].Childern, M) //get all similar child;
    DR ← Identical Data Regions (Object, M, τ);
    If coveredObjects ← (Object.Childern ∈ DR);
    foreach coveredObjects as CO do
        Remove duplicates (CO.Child.Text);
        Remove Stop Words (CO.Child.Text);
        (keys, values) = MAP(CO.Child.Text);
        KA = Reduce (Maximum TF (keys, values));
        KA = Stemming (KA)
    end
    if UncoveredObjects ← (Object.Childern ∉ DR)
    then
        Pre-processing (UncoveredObjects, M, τ)
    end
    Return DR, KA
end
```

like Apriori algorithm was applied as the first step to find a most relative keywords with high support. After that, Latent Dirichlet Allocation [32], [36] (LDA) method was applied as second step to perform topic or subject modeling. After that, fuzzy C-mean algorithm is used as third step to group related keywords and assign central keyword for each subject with a specific membership. Finally, ontology graph was created according to each topic cluster. The main goal of this part is to automatically assign the web pages with topic distributions, where a page may contain several topics that were learned with the help of statistical inference. LDA adopts the bag-of-keywords assumption since it does not take in consideration the order of the keywords in a page.

The word ontology arises from Greek ONTO (being) + LOGY (word). The subject of ontology is the study of the categories of things that exist or may exist in some domain. So, we use the ontology graph to facilitate knowledge representation and reuse. The ontology graph construction is divided into two main processes. The first process is the initial subject creation process. In this process, the graph is initially in the form of a tree structure. The first node is the main subject of the topic, according to LDA, then the related sources or objects are added in the given hierarchy, including the central keyword node as shown in fig.3. Then, to each object, the weighted keywords are added in the given hierarchy. If a keyword is a duplicate in a subject and it's objects, it is removed from the subject. To be semantically more specific, we decided to keep it as a keyword of the object. But, if a keyword is duplicated in two or more object of the same level under the same subject, it is removed from both and added to the subject.

The second process is the validation method. At this round, we have a collection of candidate keywords that require to be modified to their proper location in the graph. The issue here is about their relevance to the subject. Some of the selected keywords could be somehow far from developed subject content. Therefore, the concerned candidate keywords need to be filtered to remove irrelevant terms. The previous process is done in the Relevance validation. We start by removing the keywords that do not belong to the same category as the current node.

C. INDEXING PART (BACKEND PHASE

The part comes after the ontology creation part. In this part, outcomes from crawling, parsing and analyzing steps are stored into a database. But with massive data cannot be classified into a relational database because of scalability, storage, sorting, semantic and retrieval issues. So, Indexer block performs essential first step such as convert ontology graphs to resource description framework [34] (RDF based on XML) data model file. Then, the second step is storing RDF documents and keywords within (key, value) pairs matching to terms and documents sequentially as displayed in Fig.4.

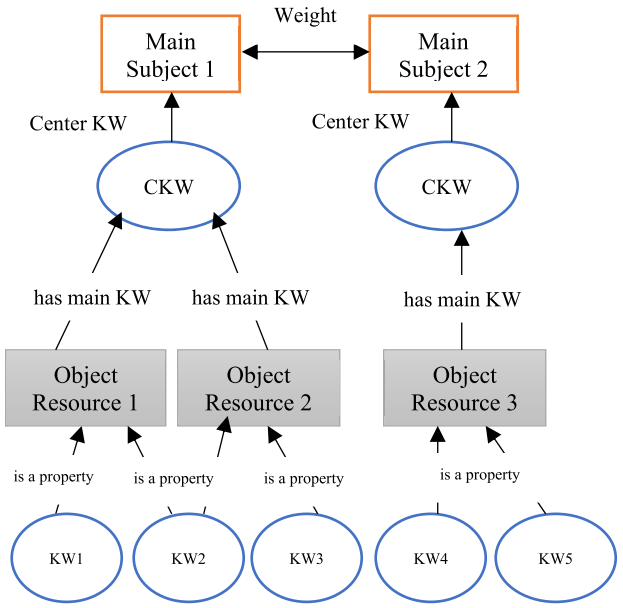

FIGURE 3. Structure of the semantic ontology graph.

Query Keyword	RDF ID
QKW_1	ID_1, ID_2, ID_5, ...
QKW_2	ID_3, ID_4, ID6, ...
...	
QKW_n	ID_i, ..., ID_{i+m}

(a)

RDF ID	Resources
ID_1	URL_i, URL_{i+1}, ..., URL_{i+N}
ID_2	URL_L, URL_{L+1}, ..., URL_{L+M}
...	
ID_n	URL_K, URL_{K+1}, ..., URL_{K+P}

(b)

FIGURE 4. Cache tables of URL resources (a) Cache table of query keywords as key and corresponding RDF Id as a value (b) Cache table of RDF Id as key and corresponding document source as a value.

The initial step shows specifically what is needed to "invert," i.e., to accumulate all values relating to the equivalent key, matching to the documents that include the equivalent term. to overcome time consuming challenge.

Due to scalability, we use NoSQL document-oriented module instead of a relational DB. Cache tables contain keywords and related RDF ID. Also, due to the semantic and unstructured problem, RDF based on XML scheme is used to support ontologies and relations.

D. RANKING PART (FRONTEND

The rank process is the main contribution of this paper and represented in Algorithm 3. All the above sections will be combined to find out the ranking score in this section. However, outcomes of crawling and indexing processes are keywords, positions, and RDF documents. We proposed a mathematical equation (3) that combines the outcomes of all the above sections that calculated in equation (1) and equation (2) respectively to find the ranking score for each document to retrieve the related document easily.

$$LW_{ij}^{in} = \frac{L_i}{\sum_{k \in j} L_k} \tag{1}$$

where LW_{ij}^{in} donates weight of link for pages i and j. Li and Lk realize the number of in links of document i and document j, sequentially. R(j) denotes the reference page list of page j.

$$CW_{xi} = \left(\frac{freq_x}{N}\right) U_{xi} \tag{2}$$

where CW_{xi} donates weight of a content for keyword x on page i. $freq_x$ donates the number of frequencies of keyword x on page i. N donates the total number of strings in page i ignoring stop words. U_{xi} donates fuzzy membership of keyword x based on cluster center page i.

$$SR_i = \sum_{x,j \epsilon i} PR_i CW_{xi} LW_{ij} \tag{3}$$

where SR (i) donates semantic rank for a document i. PR_i donates google page rank of document i.

CW_{xi} donates weight of a content for keyword x on page i calculated from equation (2). LW_{ij}^{in} donates weight of link for pages i and j calculated from equation (1).

Algorithm 3 Create Ranking Score

Input : Page (P), link Weights of P, Query (Q)
Output: Semantic Score (SS)

```
Begin Procedure;
Initially enter a Q to search.;
N = Expand Q with other keywords using wordnets;
if ∀N ∈ P then
    x ∈ N;
    Calculate CW in equation 2 for x on P;
    Find all links of P;
    Calculate LW in equation 1 for P;
    GET Page Rank (P);
    Return SS (P) in equation 3
end
End Procedure
```

E. RETRIEVAL PART (FRONTEND)

In this stage as displayed in Fig.1, three main blocks are utilized which directly connected to the user. The First block is the query engine which has a keyword generator manner that applied to divide the query call into separate words. The Second block is query optimizer which has two main substeps. Firstly, scanning the keywords claims by the end user and parsing it using algorithm#3. Secondly, optimizing the query and checks grammatical errors using the Levenshtein as character distance method.

Finally, the third block utilizing word-net to expand keywords of a given query [37]. After that, query engine compute

similarity score between expanded tags and tags in caching table. Tags with highest semantic score used to retrieve RDF documents.

The similarity score measurement is also one of the main contributions of this paper. We propose the following normalized semantic score (NSS) equations for input links in ontology graph between resource i and resource j:

$$NSS_{in}\left(R_i, R_j\right) = 1 - \left(\frac{\max\left(\log R_i, \log R_j\right) - \log\left(R_i \cap R_j\right)}{\log T - \min\left(\log R_i, \log R_j\right)}\right) \tag{4}$$

$$NSS_{in}\left(R_i, R_j\right) = \begin{cases} 0, & R_i \neq R_j \\ 1, & R_i = R_j \\ otherwise, & R_i related to R_j \end{cases} \tag{5}$$

where:

- Ri is resource i of ontology graph with assigned keyword i
- Rj is resource j of ontology graph with assigned keyword j
- T is the total resources in the grap

Fig.5 illustrates an example of frontend retrieval processing using search engine query. If we have a query (e.g. Search Engine), then the first step to split, clean and transform this query. After that, the query divided into two distinct keywords (search) and (engine). Then, every keyword was expanded using wordnet and the normalized semantic score in equation 8 was calculated. The final step, the intersection resources was summed to get the final related resources and retrieved these resources.

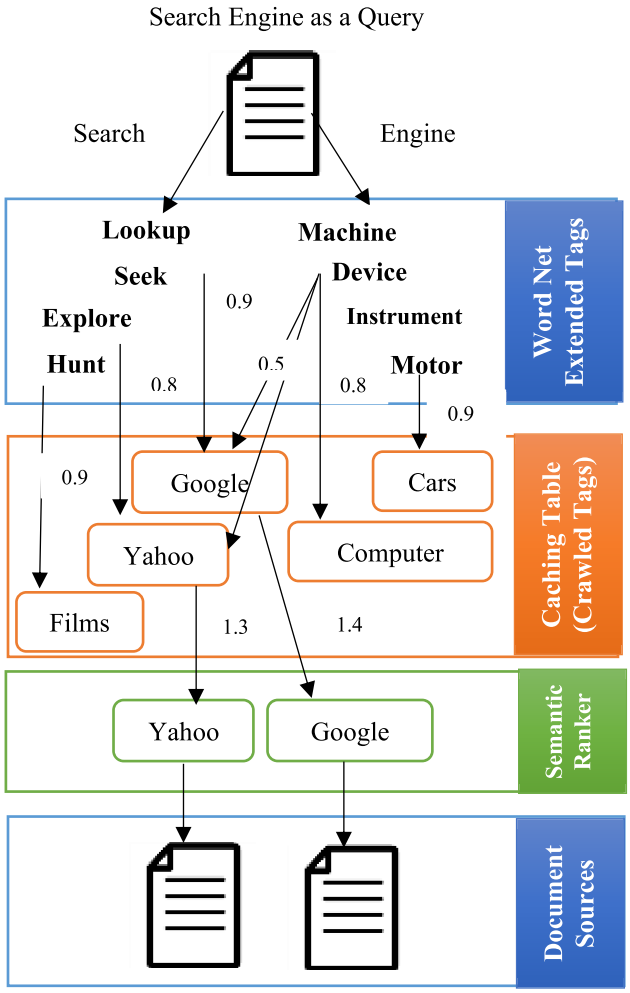

FIGURE 5. Example of frontend processing.

V. EXPERIMENTAL RESULTS

A. IMPLEMENTATION TOOLS

The recommended system has been implemented using some tools and technologies as presented in Table 1, including Apache server version 2, RDF scheme based on XML (Structure format), PHP programming language (for Sending HTTP Request, perform crawling, indexing and ranking processes), and MapReduce technology (Hadoop).

The proposed implementation runs on five personal computers which one master PC used to split tasks between slaves, and others are slaves used to perform different offline processes and PCs characteristics are Intel (R) Core (TM) i7-4702MQ CPU @ 2.20 GHz processor. PCs used to crawl datasets about 30GB of over 5 million web pages.

TABLE 1. Implementation tools.

Devices	5 PCs (1 master, 4 slaves).
Server	Apache Server version 2.
Platform	Intel (R) Core (TM) i7-4702MQ CPU @ 2.20 GHz processor.
RAM	8 GB for each PCs.
Database scheme	RDF scheme with NoSQL document-oriented module.
Programming and Query languages	PHP and SPARQL.
Size of Crawled Data	25 GB of two million web pages of different fields.

B. IMPLEMENTED TEST CASES

In this subsection, we will present different test cases using various criteria. To see if the proposed methods improve the results, we should test different queries in length using different technologies. Table 2 represents the summary of different test cases using different criteria. The implemented eight different test cases will be conducted as follows:

1) **Test Case I**: In this test case, the system is triggered by a simple unclear single word as query (e.g. engine). Also, in this test case, we will apply all different proposed techniques explained in section 4 to validate system performance and response time.
2) **Test Case II**: In this test case, the system is triggered by the same query as in test case I (e.g. engine). But, in this test case, we will apply all different proposed techniques without applying keywords expansion and similarity score based on semantic relations processes to a query.

TABLE 2. Summary of different test cases.

	Query Words	Support Ontology	Proposed Rank	Semantic Score	Keyword expansion
Test Case I	Single	Yes	Yes	Yes	Yes
Test Case II	Single	Yes	Yes	No	No
Test Case III	Single	Yes	No	Yes	Yes
Test Case IV	Single	No	No	No	No
Test Case V	Multi	Yes	Yes	Yes	Yes
Test Case VI	Multi	Yes	Yes	No	No
Test Case VII	Multi	Yes	No	Yes	Yes
Test Case VIII	Multi	No	No	No	No

3) **Test Case III**: In this test case, the system is triggered by the same query as in test case I (e.g. engine). But, in this test case, we will apply all different proposed techniques without applying proposed ranking process and caching system.
4) **Test Case IV**: In this test case, the system is triggered by the same query as in test case I (e.g. engine). The basic parts of any search engine will be applied like crawling, indexing and retrieval based on query itself. But, in this test case, we will not apply all the proposed techniques.
5) **Test Case V**: In this test case, the system is triggered by complex multiple words as query (e.g. semantic search engine). Also, in this test case, we will apply all different proposed techniques.
6) **Test Case VI**: In this test case, the system is triggered by the same query as in test case V (e.g. semantic search engine). But, in this test case, we will apply all different proposed techniques without applying keywords expansion and similarity score based on semantic relations processes to a query.
7) **Test Case VII**: In this test case, the system is triggered by the same query as in test case V (e.g. semantic search engine). But, in this test case, we will apply all different proposed techniques without applying proposed ranking process and caching system.
8) **Test Case VIII**: In this test case, the system is triggered by the same query as in test case V (e.g. semantic search engine). The basic parts of any search engine will be applied like crawling, indexing and retrieval based on query itself. But, in this test case, we will not apply all the proposed techniques.

C. EXPERIMENTAL RESULTS AND EVALUATIONS OF TEST CASES

In this sub-section, we will review the results of applying test cases mentioned in the previous subsection. It should be noted that all the results used to measure the performance of the system were calculated according to the following metrics and equations:

$$R = \frac{TP}{TP + FN} \tag{6}$$

$$P = \frac{TP}{TP + FP} \tag{7}$$

$$F - Score = \frac{2PR}{P + R} \tag{8}$$

where, TP, FP, TN, and FN which refer to true positive, false positive, true negative and false negative respectively. Recall (R) refers to the percentage of returned records that were retrieved correctly and labeled as negative as not related to the query. Precision (P) refers to the percentage of returned records that were retrieved correctly and labelled as positive as most related records to the query.

TABLE 3. Results of different test cases using single keyword in the query.

	Test Case I	Test Case II	Test Case III	Test Case 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	2.57 second	1.22 second	3.49 Second	5.4 Second
Number of Fetched Document	5300 RDF Files	10980 RDF Files	6003 RDF Files	22033 RDF Files
Methods used	Basics + All proposed methods	All Except Keyword expansion & Semantic Score	All Except Proposed Raking & Caching system	Basics only

D. DISCUSSIONS

In this sub-section, we will discuss the results of the previous subsection and infer the overall system performance. The first four test cases (from I to IV) were applied a single unclear word in the query using different techniques in each test case, as mentioned before in the previous subsection. So, the results of them were combined together in table 3 to be comparable. System performance chart for the implemented test cases from I to IV is represented in fig.6. Also, the chart of response time and number of fetched documents for the same test cases are shown in fig.7 and fig.8 respectively. The second four test

TABLE 4. Results of different test cases using multiple keywords in the query.

	Test Case V	Test Case VI	Test Case VII	Test Case VIII
Input Keywords	3	3	3	3
Extended Keywords	12	0	12	0
True Positive	4871	3982	4815	3295
False Positive	93	242	160	352
Precision	0.98	0.942	0.967	0.903
Recall	0.965	0.973	0.968	0.967
F-score	0.973	0.957	0.967	0.935
Response Time at Online Phase	1.78 second	0.88 second	2.18 Second	3.34 Second
Number of Fetched Document	2309 RDF Files	6400 RDF Files	4055 RDF Files	11015 RDF Files
Methods used	Basics + All proposed methods	All Except Keyword expansion & Semantic Score	All Except Proposed Raking & Caching system	Basics only

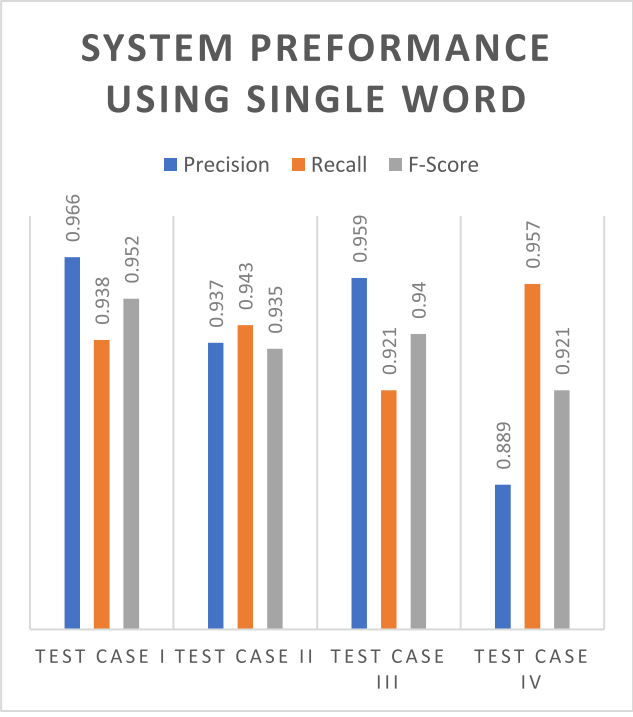

FIGURE 6. System performance chart for test cases from I to IV that applied a single word in the query.

cases (from V to VIII) were applied multiple words with some relation between them in the query using different techniques in each test case. So, the results of them were combined

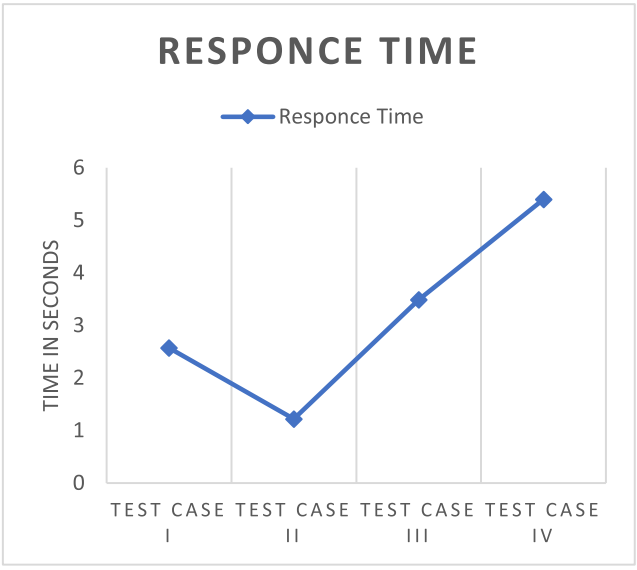

FIGURE 7. Response time chart for test cases from I to IV that applied a single word in the query.

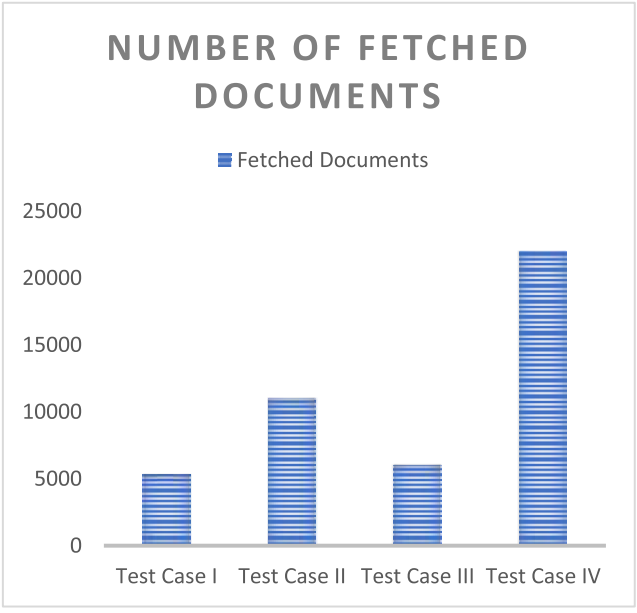

FIGURE 8. Number of fetched documents chart for test cases from I to IV that applied a single word in the query.

together in table 4 to be comparable. Also, System performance chart for the implemented test cases from V to VIII is represented in fig.9. In addition, the chart of response time and number of fetched documents for the same test cases are shown in fig.10 and fig.11 respectively. From table 3, table 4 and the corresponding charts from 6 to 11, we can infer the following information:

- Test case V achieved the highest precision rate among all test cases because the number of keywords that entered and extracted were increased.
- Test case I and test case V achieved a higher precision rate than other related test cases because all proposed methods were applied together.

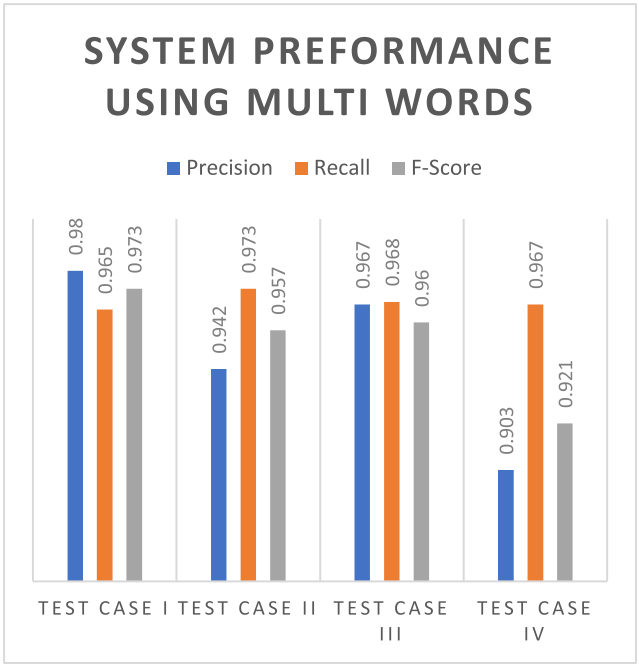

FIGURE 9. System performance chart for test cases from V to VIII that applied multi words in the query.

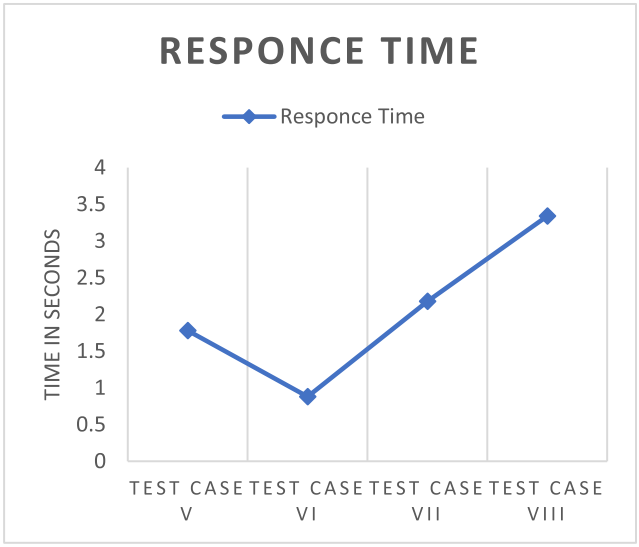

FIGURE 10. Response time chart for test cases from V to VIII that applied multi words in the query.

- Test case III, test case IV, test case VII and test case VIII achieved a higher time consumption than other first four test cases because the caching system and ranking process were not applied in this test case.
- Test case IV and test case VIII fetched the highest number of RDF documents because the ranking, ontology, semantic score and keyword expansion processes were not applied.
- The more keywords user entered in the query, the more keywords can be extracted and more relationships can be created.

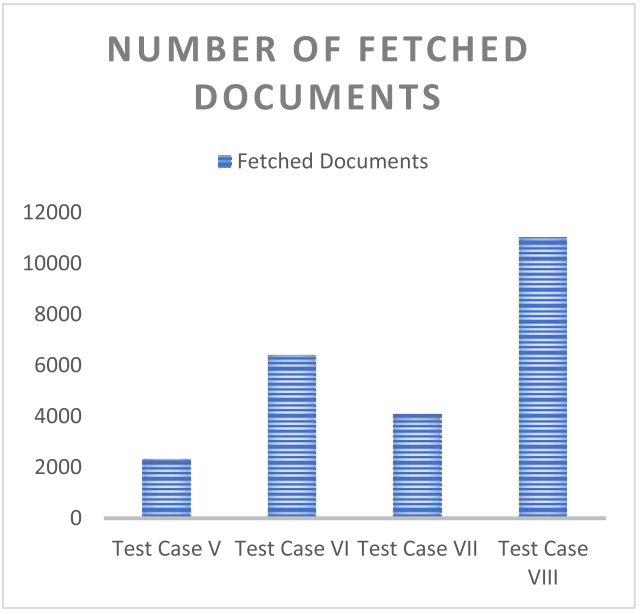

FIGURE 11. Number of fetch documents chart for test cases from V to VIII that applied a single word in the query.

TABLE 5. Comparison between the proposed ranking method and other ranking methods in different systems.

Ranking method	Weighted Page Rank	Distance Ranking Method	Hits Ranking method	Proposed Ranking method
Type	Text Ranker	Text Ranker	Image Ranker	Text Ranker
Mining method	Web Structure and content	Web Structure	Web content	Web Structure and some Content
Cons	Not Efficient	Weak with Big data	Speed Problem	Work with Page Rank
Factors	visited links	Back links only	Image Content	Backlink, title and page meta data
Speed	Fast	Medium	Slow	Medium

- The more keywords user entered in the query, the fewer documents that have been fetched.

E. COMPARATIVE STUDY BETWEEN PROPOSED RANKING ALGORITHM AND OTHER RANKING METHODS

In this sub-section, we conduct a comparison between proposed ranking procedure and other related ranking procedures of web pages. Each procedure has its benefits and drawbacks. The procedure which comes under the diverse types is compared under some factors like type of search, parameters, mining procedure, limitation, and

TABLE 6. Comparison between different systems using test case I (simple unclear query).

	Google System	Swoogle System	Falcon System	Proposed System
Precision	0.93	0.95	0.94	0.97
Recall	0.98	0.90	0.93	0.93
F-measure	0.955	0.925	0.935	0.95
Retrieved Data	20M	20K	5K	10K
Avg. Response Time	1.53	3.11	2.92	2.57

TABLE 7. Comparison between different systems using test case V (Complex query).

	Google System	Swoogle System	Falcon System	Proposed System
Precision	0.95	0.96	0.95	0.98
Recall	0.98	0.94	0.95	0.96
F-measure	0.96	0.95	0.95	0.97
Retrieved Data	2M	13K	Less 1K	8K
Avg. Response Time	0.65	2.44	2.15	1.78

TABLE 8. Comparison between the proposed system and other related systems.

	Google	Swoogle	Falcon	Proposed
Support Ontology	×	√	√	√
Semantic Relations	×	√	√	√
Support Ranking	√	×	√	√
Handle Big Data	√	×	×	√
High Precision	×	√	√	√
High Recall	√	×	×	×
Response Time	Fast	Medium	Slow	Medium

performance. Table 5 shows the assessment of diverse procedures against the proposed procedure based on different criteria.

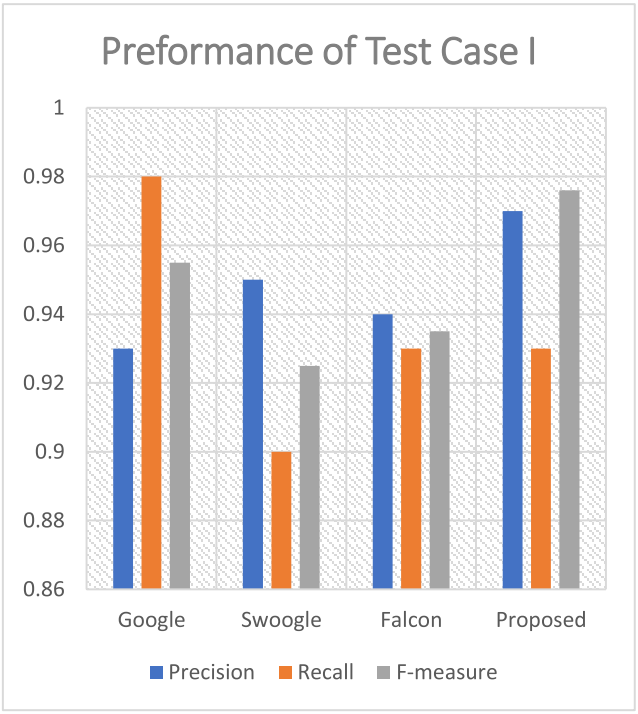

FIGURE 12. Performance chart of different systems using test case I.

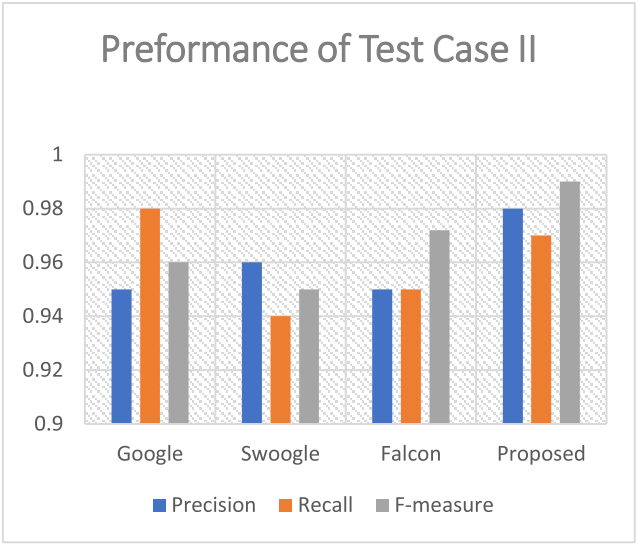

FIGURE 13. Performance chart of different systems for test case V.

F. COMPARATIVE STUDY BETWEEN PROPOSED SYSTEM AND OTHER RELATED SYSTEMS

Also, we conduct a comparison between the proposed system and other related systems in the same field. Every system has its pros and cons. The comparison comes under the different criteria like supporting semantic, handling big data, support ontologies, performance and time-consuming. The test cases I and V were conducted famous search system such as Google, Swoogle, falcon and proposed system. Test cases I and V were used in the comparison with other related systems because they uses all proposed techniques and they

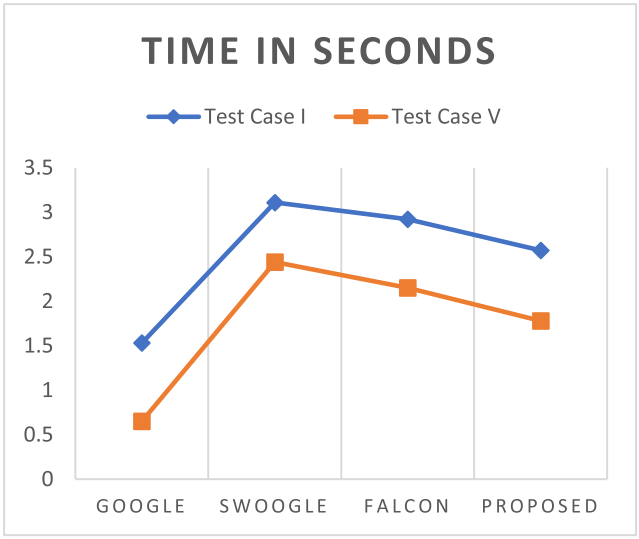

FIGURE 14. Average elapsed time of different systems using different test cases.

achieved the highest accuracy among test cases in previous sub-section. Experimental outcomes with averages were conducted and represented in table 6 and table 7 for test cases I and V respectively. We can infer from these outcomes that the proposed system achieved high precision than other related systems. Table 8 shows the assessment of diverse related systems against the proposed system of web search based on different criteria.

VI. CONCLUSION

This paper proposes an information retrieval framework for extracting information from the recorded data based on ontology annotations and a mathematical ranking model based on semantic associations. Experimental results showed high precision and accuracy of our test cases in minimum time comparing to related systems. Experiments with eight different test cases are conducted to evaluate the proposed system. These test cases show that our model obtains comparable accuracy with high precision 97% in good response time comparing to related systems.

REFERENCES

[1] G. Sudeepthi, G. Anuradha, and M. S. Babu, "A survey on semantic Web search engine," *Int. J. Comput. Sci. Issues*, vol. 9, no. 2, pp. 241–245, Mar. 2012.
[2] R. Sachdeva and S. Gupta, "A literature survey on page rank algorithm," *Int. J. Sci. Eng. Res.*, vol. 9, no. 5, pp. 10–19, May 2018.
[3] H. Dong, F. K. Hussain, and E. Chang, "A survey in semantic search technologies," in *Proc. 2nd IEEE Int. Conf. Digit. Ecosyst. Technol.*, Feb. 2008, pp. 403–408.
[4] J. M. Kassim and M. Rahmany, "Introduction to semantic search engine," in *Proc. Int. Conf. Elect. Eng. Inform.*, vol. 2, Aug. 2009, pp. 380–386.
[5] S. Khusro and I. Ullah, "Towards a semantic book search engine," in *Proc. Int. Conf. Open Source Syst. Technol. (ICOSST)*, Dec. 2016, pp. 106–113.
[6] D. K. Sharma, "A comparative analysis of Web page ranking algorithms," *Int. J. Comput. Sci. Eng.*, vol. 2, no. 8, pp. 2670–2676, Aug. 2010.
[7] R. Sharma, A. Kandpal, P. Bhakuni, R. Chauhan, R. H. Goudar, and A. Tyagi, "Web page indexing through page ranking for effective semantic search," in *Proc. 7th Int. Conf. Intell. Syst. Control (ISCO)*, Jan. 2013, pp. 389–392.
[8] M. M. El-gayar, N. Mekky, and A. Atwan, "Efficient proposed framework for semantic search engine using new semantic ranking algorithm," *Int. J. Adv. Comput. Sci. Appl.*, vol. 6, no. 8, pp. 136–143, Aug. 2015.
[9] A. Fatima, C. Luca, and G. Wilson, "New framework for semantic search engine," in *Proc. UKSim-AMSS 16th Int. Conf. Comput. Modelling Simulation*, Mar. 2014, pp. 446–451.
[10] P. Kolle, S. Bhagat, S. Zade, B. Dand, and C. S. Lifna, "Ontology based domain dictionary," in *Proc. Int. Conf. Smart City Emerg. Technol. (ICSCET)*, Jan. 2018, pp. 1–4.
[11] V. Jain and M. Singh, "Ontology based information retrieval in semantic Web: A survey," *Int. J. Inf. Technol. Comput. Sci.*, vol. 5, no. 10, pp. 62–69, Sep. 2013.
[12] A. A. Khan and S. K. Malik, "Semantic search revisited," in *Proc. 8th Int. Conf. Cloud Comput., Data Sci. Eng. (Confluence)*, Jan. 2018, pp. 14–15.
[13] P. Banik, S. Gaikwad, A. Awate, S. Shaikh, P. Gunjgur, and P. Padiya, "Semantic analysis of wikipedia documents using ontology," in *Proc. IEEE Int. Conf. Syst., Comput., Automat. Netw. (ICSCA)*, Jul. 2018, pp. 1–6.
[14] 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.
[15] R. Jain, N. Duhan, and A. K. Sharma, "Comparative study on semantic search engines," *Int. J. Comput. Appl.*, vol. 131, no. 14, pp. 4–11, Dec. 2015.
[16] M. Klusch, P. Kapahnke, S. Schulte, F. Lecue, and A. Bernstein, "Semantic Web service search: A brief survey," *Künstliche Intell.*, vol. 30, no. 2, pp. 139–147, 2016.
[17] L. Ding, T. Finin, A. Joshi, and R. Pan, "Swoogle: A semantic Web search and metadata engine," in *Proc. 13th ACM Conf. Inf. Knowl. Manage.*, 2004, pp. 1110–1145.
[18] 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 Semantics*, vol. 9, no. 4, pp. 365–401, Dec. 2011.
[19] Y. Qu and G. Cheng, "Falcons concept search: A practical search engine for Web ontologies," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 41, no. 4, pp. 810–816, Jul. 2011.
[20] L. Laura and G. Me, "Searching the Web for illegal content: The anatomy of a semantic search engine," *Soft Comput.*, vol. 21, no. 5, pp. 1245–1252, Mar. 2015.
[21] M. Al-Yahya, H. Al-Khalifa, A. Bahanshal, I. Al-Odah, and N. Al-Helwah, "An ontological model for representing semantic lexicons: An application on time nouns in the holy quran," *Arabian J. Sci. Eng.*, vol. 35, no. 2, p. 23, Dec. 2010.
[22] L. Al-Safadi, D. Al-Rgebh, and W. AlOhali, "A comparison between ontology-based and translation-based semantic search engines for arabic blogs," *Arabian J. Sci. Eng.*, vol. 38, no. 11, pp. 2985–2992, Nov. 2013.
[23] W. Medhat, K. M. Fouad, A. H. Yousef, and I. F. Moawad, "An arabic semantic search engine for large governmental organization," in *Proc. 12th Int. Conf. Comput. Eng. Syst. (ICCES)*, Dec. 2017, pp. 564–570.
[24] S. S. Laddha and P. M. Jawandhiya, "Semantic Search Engine," *Indian J. Sci. Technol.*, vol. 10, no. 21, pp. 1–6, 2017.
[25] A. Abatal, H. Khallouki, and M. Bahaj, "A semantic smart interconnected healthcare system using ontology and cloud computing," in *Proc. 4th Int. Conf. Optim. Appl. (ICOA)*, Apr. 2018, pp. 1–5.
[26] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the Web," Stanford Digit. Libraries, Dept. Comput. Sci., Stanford Univ., Tech. Rep. SIDL-WP-1999-0120, 1999, pp. 1–17.
[27] N. Duhan, A. K. Sharma, and K. K. Bhatia, "Page ranking algorithms: A survey," in *Proc. IEEE Int. Advance Comput. Conf.*, Mar. 2009, pp. 1530–1537.
[28] N. Tyagi and S. Sharma, "Weighted page rank algorithm based on number of visits of links of Web page," *Int. J. Soft Comput. Eng.*, vol. 2, no. 3, pp. 2231–2307, Jul. 2012.
[29] S. Elbassuoni, M. Ramanath, R. Schenkel, and G. Weikum, "Searching RDF graphs with SPARQL and keywords," *IEEE Data Eng. Bull.*, vol. 33, no. 1, pp. 16–24, Mar. 2010.
[30] D. Bollegala, Y. Matsuo, and M. Ishizuka, "A Web search engine-based approach to measure semantic similarity between words," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 7, pp. 977–990, Jul. 2011.
[31] M. Lenzerini, "Ontology-based data management," in *Proc. 20th ACM Int. Conf. Inf. Knowl. Manage.*, vol. 866, Oct. 2011, pp. 12–15.
[32] F. Gurcan and N. E. Cagiltay, "Big data software engineering: Analysis of knowledge domains and skill sets using LDA-based topic modeling," *IEEE Access*, vol. 7, pp. 82541–82552, 2019.

[33] A. Oussous, F. Z. Benjelloun, A. A. Lahcen, and S. Belfkih, "Big data technologies: A survey," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 30, no. 4, pp. 431–448, Oct. 2018.
[34] Q. Tong, "Mapping object-oriented database models into RDF(S)," *IEEE Access*, vol. 6, pp. 47125–47130, 2018.
[35] E. L. Lydia and M. Ben Swarup, "Analysis of big data through Hadoop ecosystem components like flume, mapreduce, pig and hive," *Int. J. Comput. Sci. Eng.*, vol. 5, no. 1, pp. 21–29, 2016.
[36] P. Anupriya and S. Karpagavalli, "LDA based topic modeling of journal abstracts," in *Proc. Int. Conf. Adv. Comput. Commun. Syst.*, Jan. 2015, pp. 1–5.
[37] M. G. Buey, L. Garrido, and S. Ilarri, *An Approach for Automatic Query Expansion Based on NLP and Semantics* Cham, Switzerland: Springer, 2014, pp. 349–356.

M. M. EL-GAYAR received the B.Sc. and M.Sc. degrees in information technology from Mansoura University, Mansoura, Egypt, in 2010 and 2013, respectively, where he is currently pursuing the Ph.D. degree with the Faculty of Computer and Information Science.

Since 2013, he has been a Teaching Assistant with the Faculty of Computer and Information Science, Mansoura University. His research interests include semantic web, algorithms, big data, and pattern recognition.

N. E. MEKKY received the B.Sc., M.Sc., and Ph.D. degrees in electronics engineering from Mansoura University, Mansoura, Egypt.

She was a Researcher with Department of Electronics Communication Engineering, Mansoura University, for seven years, before joining the Misr Higher Institute for Engineering and Technology, Mansoura, in 2007, as an Assistant Lecturer. She is currently an Assistant Professor with the Department of Information Technology, Faculty of Computers and Information Systems, Mansoura University. Her research interests include image processing, semantic web, the IOT, and biomedical engineering.

A. ATWAN received the B.Sc., M.Sc., and Ph.D. degrees in electronics and communications engineering from the Faculty of Engineering, Mansoura University, Egypt, in 1988, 1998, and 2004, respectively.

From 2005 to 2009, he was an Assistant Professor with the Faculty of Computers and Information Sciences (FCIS), Mansoura University, where he was an Associate Professor, from 2010 to 2015, and has been a Professor, since 2016. Since 2019, he has been a Professor with the Faculty of computer and information Technology, Northern Border University, Saudi Arabia. His research interests include semantic web, pattern recognition, networking, signal processing, and image processing.

H. SOLIMAN received the B.Sc., M.Sc., and Ph.D. degrees in electronics and communications engineering from the Faculty of Engineering, Mansoura University, Egypt, in 1983, 1987, and 1993, respectively.

From 2000 to 2012, he was an Associate Professor with the Faculty of Engineering, Mansoura University, where has been a Professor and the Dean of the Faculty of Computer and Information Science (FCIS), since 2012 and 2016, respectively. His research interests include semantic web, pattern recognition, communication, networking, and parallel computing.