International Journal of Grid and Distributed Computing
Vol. 11, No. 7 (2018), pp.63-80
http://dx.doi.org/10.14257/ijgdc.2018.11.7.07

Semantic Indexing of a Corpus

Madani Youness[1*], Erritali Mohammed[2] and Bengourram Jamaa[3]

[1,2,3]Sultan Moulay Slimane University, Faculty of Sciences and Techniques Beni Mellal, Morocco
[1]younesmadani9@gmail.com, [2]m.erritali@usms.ma, [3]bengoram@yahoo.fr

Abstract

The field of information retrieval (IR) is an important area in computer science, this domain helps us to find information that we are interested in from an important volume of information. Recently a lot of works use semantics in information retrieval systems (IRS) to find the information easily, in this paper our job is to propose a new approach to calculate the semantic similarity between documents, our approach consists of making a hybridization between two approaches already existing in the literature, using semantic relationships such as synonymy or generalization, based on the lexical-semantic dictionary WORDNET.

We apply our approach in a first step to make a semantic IRS working in a simple system with a single server, and in a second step to calculate the semantic similarity between documents in a Big Data system using the Hadoop framework working with the MapReduce programming model and the Hadoop Distributed file system HDFS.

Keywords: *information retrieval (IR); information retrieval systems (IRS); semantic similarity; hybridization; WORDNET; Big Data; Hadoop; MapReduce; HDFS*

1. Introduction

Today we are witnessing a constant development of information technology. These new technologies have allowed the rapid evolution of technology and information management. The progress of information production tools such as text editors allowed daily production of an enormous mass of information. The evolution of electronic media allowed the storage's problem of this vast amount of information. This rapid increase in the volume of information has created the problem of how to find an information that interests us in this great mass of information. To address this problem an entire discipline was born. This discipline is known as Information Retrieval (IR) [1].

To do this search we need research tools that are called Information Retrieval Systems (IRS). Thus, an IRS can select from a volume of information, relevant information for user's need. In these systems, the need of information is expressed by a query.

Each IRS involves two important phases, the indexing phase, that is to say, each document is represented by an intermediate representation. The IRS directly operates this representation. It describes the document content by descriptors. These descriptors are meaningful units in the document. This description is called the index of the document, in the same way; the content of the query is described by a set of descriptors. The second phase is the phase of researching the relevant documents to a user need (query). To find the relevant documents for a query, the IRS compares the performance of that query to the representation of each document. This comparison

Received (March 26, 2018), Review Result (June 4, 2018), Accepted (June 14, 2018)
* Corresponding Author

ISSN: 2005-4262 IJGDC

is executed using a mapping function (Retrieval Status Value: RSV) [2] and a relevance score is assigned to each document. These scores can present the documents in order of relevance.

The classical IRS, treat the documents as sets of words, called bag of words. These words are used in these IRS to describe the content of a document. Thus, these IRS consider words like writing without meaning. Therefore, they can only find documents that are described by the same words as the query. For example, a document indexed by a word which is a synonym of another word that describes the query will never be returned by these IRS, although this document is relevant. To overcome these limitations it has become essential to consider the meaning of the words. The descriptors are then the meaning of words: concepts. This type of indexing is called conceptual or semantic indexing.

Indexing by keywords is generally unclear. This imprecision is due to semantic ambiguity problem of natural language words. For example, the same word can have several meanings and different words can have different meanings. Therefore, although relevant documents contain words semantically equivalents but lexically different (synonyms) to the query words will not be found. Furthermore, an irrelevant material containing words lexically identical but semantically different (homonyms) to the words of the query will be returned to the user.

The solution to this problem is the semantic indexing of a corpus, that is to say, the use of semantic relations in the phase of indexing and retrieving of documents. It is in this sense that our approach is situated, it allows the use of semantic relations such as synonymy and some approaches already existing in the literature that use semantics.

We can distinguish three main approaches for the similarity calculation between the taxonomy objects. The first type is based on the nodes[3][4]. Works under the banner of these approaches used the typically information based content to determine the conceptual similarity. Moreover, the similarity between two concepts is obtained by the degree of sharing information. The second type is based only on the hierarchy or the edge distances [5][6]. The problem with this approaches is that the taxonomy arcs represent uniform distances, *i.e.*, all the semantic links have the same weight. Finally, the hybrid approach [7][8] which combines the two approaches presented above. With these approaches, there exist several manners of detecting the conceptual similarity of two words in a hierarchical semantic network.

Our approach is to hybridize two representation methods, the first is the vector representation of documents using the relation of synonymy, and the second is to use an approach already existing in the literature.

A problem arises in the IRS that contain large databases (large corpus), is the increased time for the indexing and the search stages, the solution to this problem is to parallelize our approach of calculating the similarity between a query and each document of the corpus, working in a big data system[9], by sharing the work of indexing and calculation of the similarity between several machines.

In this article, we present the different experimental results obtained by the application of our approach to build an IRS in a simple system and to calculate the semantic similarity between documents in a distributed system (Big Data).

2. Related Work

Many studies have been presented on detecting document similarity in recent years for facilitating the search of information in complex information systems. Kumar *et al.*, [10] and Chowdhury[11] surveyed duplicate or near duplicate data detection algorithms. Related work on text similarity detection can be mainly classified into two categories: traditional method and parallel method.

For the traditional methods, Lyon *et al.*, [12] proposed a tri-gram and set theory-based algorithm, a data finger-based method, to extract the data finger of sentences and then mapped it into a range of value using Hash or MD5 function, then, reported the similarity according to the overlapped ratio of similar value or the maximum common sub-sequence.

For the parallel-based methods, most approaches focused on MapReduce model. Zhang *et al.*, [13] presented a sequence-based method to detect the partial similarity of a web page using MapReduce, which consisted of two sub-tasks as sentence-level near-duplicate detection and sequence matching. In this work, will be also used MapReduce framework, but we integrate it with some effective features to guarantee running time performance.

3. Application Areas of Similarity

To motivate the importance of the similarity measure, we summarize its uses in several application areas.

3.1. Language Automatic Treatment

Several works to the similarity measure were justified by the language automatic treatment (LAT). Among works in this domain we can cite the work of [14] which uses metric semantic similarity to measure the semantic similarity between all the word sense of a given words pair and disambiguate thus in a given context. [15] Combined the use of a thesaurus automatically acquired starting from the rough textual corpora and WordNet (based on the metric of the similarity) to find sense prevalent of the words in not structured texts. The authors of work [16] applied semantic similarity measures of WordNet to evaluate the expressions relevance, being given a specific dialogue, and automatically to build the synopses of the spoken dialogue. In this same domain, you can consult work of [17, 18].

3.2. Bioinformatics

The large-scale effort in developing, maintaining and making biomedical ontology available motivates the application of similarity measures to compare ontology concepts or, by extension, the entities described therein. A common approach, known as semantic similarity, compares ontology concepts through the information content they share in the ontology. Ontology-based similarity has become a prominent approach to compare biomedical entities based on their biomedical activity. A variation of similarity measure based on the informational contents is adapted to find a better way of organizing and questioning the Gene Ontology data (GO) [19].

3.3. Information Retrieval

Information retrieval rests largely on measures for the identification of the similarity between the documents [6,9]. The majority of the approaches of the information research takes into account only simple words and/or fragments of the words for the research of documents and is unaware of the essential idea that takes into account the ontological ratios of the words. The latter can be detected by a computing process of similarity between pairs of objects.

3.4. Web Services

Determining the similarity of semantic services provides useful information regarding their accountabilities. In the work of [20], there is a proposal metric to measure the similarity of semantic services annotated with an OWL ontology. The

similarity measure proposed is based on intuition than similar objects share the most descriptive information municipalities.

3.5. Detection Links

In the work [21], there is a description of the improved link detection based systems history using specific information source and combining a number of similarity measures. The similarities of the measures adopted by this work are represented by cosine, Hellinger, Tanimoto and clarity. Each of these measures captures different aspects of similarity of words in a document.

4. Classification of Semantic Similarity Measurement Approaches

In this section, we present the approaches of semantic similarity already existing in the literature, which are classified into three categories.

4.1. Approaches based on Arcs (distances)

The majority of similarity measurement of concepts in the ontology is based on their distances [5, 6]. Obviously, the concept X is more similar to a concept Y than a concept Z, this similarity is evaluated by the distance, which separates the concepts in the ontology. These measurements make use of the hierarchical structure of ontology to determine the semantic similarity between the concepts.

Among the works classified under this banner include:

4.1.1. Rada et al. Measurement

This measure[5] is adopted in a semantic network and it is based on the fact that we can compute the similarity based on the links hierarchical (generalization) "is-a". To compute the similarity of two concepts in an ontology, we must calculate the number of minimum Arcs that separate them. This measure is based on the computation of the distance between the nodes by the shortest path. The similarity measure with this measurement between the concept c_1 and the concept c_2 is presented in the following formula:

$$Sim_{Rada}(c_1, c_2) = \frac{1}{1 + dist(c_1, c_2)} \quad (1)$$

$dist(c_1, c_2)=Min_{chemin}(c_1, c_2)$ is the shortest path between the concept c_1 and the concept c_2.

4.1.2. Wu and Palmer Measurement

The principle of this measurement [6] is: given an ontology formed by a set of nodes and a root node (R) (Figure 1). X and Y represent two ontology elements for which we will compute the similarity. The principle of this similarity measurement is based on the distances (N_1 and N_2) which separate the X and Y nodes from the node R and the distance (N) which separates the Subsuming Concept (SC).

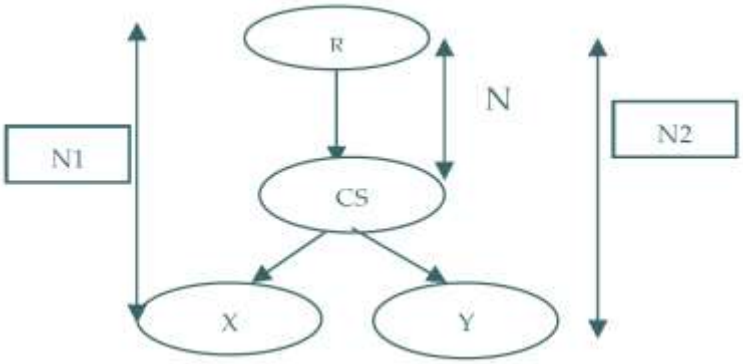

Figure 1. Example of an Extract of Ontology

The Wu and Palmer measurement is defined by this formula:

$$Sim_{Wu\&Palmer} = \frac{2 * N}{N_1 + N_1} \tag{2}$$

4.2. Node-based Approaches (Information Content)

These techniques adopt a new measure in terms of the entropic measurement of the information theory [3][4]. The probability P (.) for the identification of the use of a class or its descendants in a corpus indicates the class information. The following formula defines the entropy of a class:

$$E(c) = -\log(P(c)) \tag{3}$$

Where P is the probability of finding a concept value of c. the probability of concept c is calculated by dividing the values number of c by the full number of the values.

4.2.1. Resnik Measure

The notion of the Informational Contents (IC) was initially introduced by [4], which proved that an object (word) is defined by the number of the specified classes and that the semantic similarity between two concepts is measured by the quantity of information which they share. The informational contents are obtained by computing the object frequency in the corpus. The formula for this measure is:

$$Sim(c_1, c_2) = Max[E(CS(c_1, c_2))] = Max[-\log(p(CS(c_1, c_2)))] \tag{4}$$

$CS(c_1,c_2)$: represents the most specific concept (which maximizes the similarity value) between the concept c_1 and c_2 in the on tology.

4.2.2. Lin Measure

Lin [3] has defined a different similarity measure than that of Resnik by this formula:

$$Sim_{Lin} = \frac{2 * \log(P(CS(c_1, c_2)))}{\log(P(c_1)) + \log(P(c_1))} \tag{5}$$

This measure uses a hybrid approach that combines two different sources of knowledge (Thesaurus, corpus).

4.3. Hybrid Approches

These techniques are regarded as a factor of decision. Founded on a model that combines between the approaches based on the arcs (distances) in addition to the informational contents that are regarded as a factor of decision.

4.3.1. Jiang and Conrath Measure

To cure the problem presented to the level of the Resnik measurement, Jiac [7] brought a new formula which consists in combining the Entropy (Informational Contents) of the specific concept to those of the concepts which we seek the similarity. This approach is computed by the following formula:

$$Sim_{jiac}(c_1, c_2) = \frac{1}{distance(c_1, c_2)} \tag{6}$$

The distance between c_1 and c_2 is computed by the following formula:

$$distance(c_1, c_2) = \mathrm{E}(c_1) + \mathrm{E}(c_2) - (2 * \mathrm{E}(\mathrm{CS}(c_1, c_2))) \tag{7}$$

4.3.2. Leacock and Chodorow Measure

Another method presented by [8], which combines between counting of the arcs method and the informational content methods. The proposed measure by Leacock and Chodorow is based on the shortest path between two synsets of Wordnet. This technique is defined by the following formula:

$$Sim_{lc} = -\log(\frac{cd(c_1, c_2)}{2 * \mathrm{M}}) \tag{8}$$

M is the longest path, which separates the concept root, of the ontology, from the concept in the bottom. We indicate that cd(c_1, c_2) is the shortest path that separates c_2 of c_2.

4.4. Approaches Based on the Vector Space

These approaches use a characteristic vector, in a dimensional space, to represent each object and calculate the similarity. The similarity definition between two vectors of objects is obtained by their internal contents. Here are some approaches mentioned in the literature:

4.4.1. Jaccard Similarity

It is defined by the common objects number divided by the objects full number minus the common objects number:

$$sim_{jacard}(\mathrm{X}, \mathrm{Y}) = \frac{\sum_{i=1}^{n} x.y}{\sqrt{(\sum_{i=1}^{n} x^2)} + \sqrt{(\sum_{i=1}^{n} y^2)} - \sum_{i=1}^{n} x.y} \tag{9}$$

4.4.2. Cosine Similarity

It uses the complete vector representation, that is to say the objects frequency (words). Two documents are similar if their vectors are combined. If two objects are not similar, their vectors form an angle (X, Y) whose Cosine represents the similarity

value. The formula is defined by the ratio of the scalar product of vectors X and Y and the product of the norm of x and y.

$$Sim_{cos}(X,Y) = \frac{\sum_{i=1}^{n} x.y}{\sqrt{(\sum_{i=1}^{n} x^2)}.\sqrt{(\sum_{i=1}^{n} y^2)}} \quad (10)$$

The measurement of Cosine quantifies the similarity between the two vectors as the cosine of the angle between two vectors.

4.4.3. Euclidean Measure

The Euclidean similarity is based on the ratio of the Euclidean distance increased by one. The Euclidean distance is defined by the following formula:

$$dE = \sqrt{\sum_{i=1}^{n} (x-y)^2} \quad (11)$$

The similarity measure is therefore defined by:

$$Sim_{euclide}(X,Y) = \frac{1}{1+dE} \quad (12)$$

5. The Proposed Similarity Measure

Our approach is based on a hybrid representation that combines two methods to calculate the semantic similarity between two documents.

- **The First Method:**

The first method is based on the vector representation of documents in which each element represents the weight of a word in the document that is to say the frequency of words which equal to the number of occurrences of the word in the document.

After that our proposal consists in enriching the vector representation of each document with a semantic relationship. The idea is that for each word of the document we calculate its frequency and also the frequency of its synonyms using in this stage the WordNet dictionary based on the synset (a group of interchangeable words, denoting a sense or an individual use [22]). That is to say that the frequency of a word is equal to the sum of its frequency and that of its synonyms.

- **The Second Method:**

The second method is based on one of the approaches presented in the last section. After indexing each document, we can calculate the semantic similarity between two documents based on these approaches, but this time not only between words but between documents which contain a set of words.

Our job is to add two new things to calculate the semantic similarity between two documents, the first is to use of the semantic relation of synonymy in the indexing phase particularly in the phase of calculation of the weight of words and the second is to use one of the approaches presented in the last section.

Our proposed approach to calculate the semantic similarity between two documents (d and q), with $d=(d_1,d_2,\ldots\ldots,d_n)$ and $q=(q_1,q_2\ldots\ldots,q_n)$ are vector representations, is defined by the following formula:

$$Sim(q,d) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} q_i d_j sim(i,j)}{\sum_{i=1}^{n}\sum_{j=1}^{n} q_i d_j} \quad (13)$$

With:

- i: represents a word of the document q.
- j: represents a word of the document d.
- qi is the weight of the word i and its synonyms in the document q.
- dj is the weight of the word j and its synonyms in the document d.
- Sim(i, j) is the semantic similarity between the word i and j, calculated using one of the approaches mentioned in the last section.

5.1. Choice of the Approach

The choice of an approach from those presented in the last section is very important in our approach because it plays an important role in the research. To facilitate the choice of an approach than another, we try to calculate the semantic similarity between identical documents and choose the approaches that give good results, Table 1 shows the results.

Table 1. Similarity and Run Time in msec with Different Approaches

Measure / approach	Similarity	Time(msec)
Leacock and Chodorow	0.14	1016
Wu and Palmer	0.13	1297
Resnik	0.04	1360
JiangConrath	0.04	1391
Lin	0.02	1344

From this table, we remark that the best approach is that of Leacock and Chodorow that gives the greatest similarity with a minimum run time also the approach of wu palmer gives good results.

So from these results, we choose to work in our work with the approach of **Leacock and Chodorow** or the **Wu and Palmer** approach.

5.2. Why we use the Synonymy to Calculate the Weight of Words?

To demonstrate the utility of using the synonymy relationship in our approach when calculating the weight of words in each document, we do an experience allowing calculating the semantic similarity between a document and itself by hybridizing it with the approach of Learock and Chodorow. Table 2 illustrates the results obtained.

Table 2. Results of Similarity with and Without Synonymy

Measure / Method	Similarity
with synonymy	0.21
without synonymy	0.12

From this table we see the good effect of using the synonymy when calculating the frequency of words, so with the use of synonymy, the semantic similarity is increased

almost the twice than in the case where we do not use it, that shown the importance of using semantic relations especially the synonymy.

6. Evaluation of our Approach

In this section, we present the practical results of our approach compared with existing approaches such as **Wu and Palmer** and **Leacock and Chodorow**.

Table 3 shows the results of the similarity and the execution time between two identical documents using our approach, the approach of Wu and Palmer and the approach of Leacock and Chodorow.

Table 3. Similarity and Execution Time of our Approach Compared with WP and LC

Wu and Palmer(WP)	0.13	1297
our approche with WP	0.25	2469
Learock and chodorow(LC)	0.14	1016
our approche with LC	0.26	2329

From this table we see that despite our approach has a greater run time than Wu and Palmer or Learock and Chodorow but it gives a great similarity than the other two approaches, and because we calculate the similarity between a document and itself, our approach gives good results and it provides an important similarity than the other two approaches (Wu and Palmer or learock and Chodorow).

We can conclude from these practical results that our approach is effective to an information retrieval system because it can easily give us semantically similar documents for a query.

Figure 2 shows the results of applying our approach to calculate the semantic similarity between a query, and a corpus that contains each time a variable number of documents; and calculating each time the time needed to find the similarity between the query and any document of the corpus.

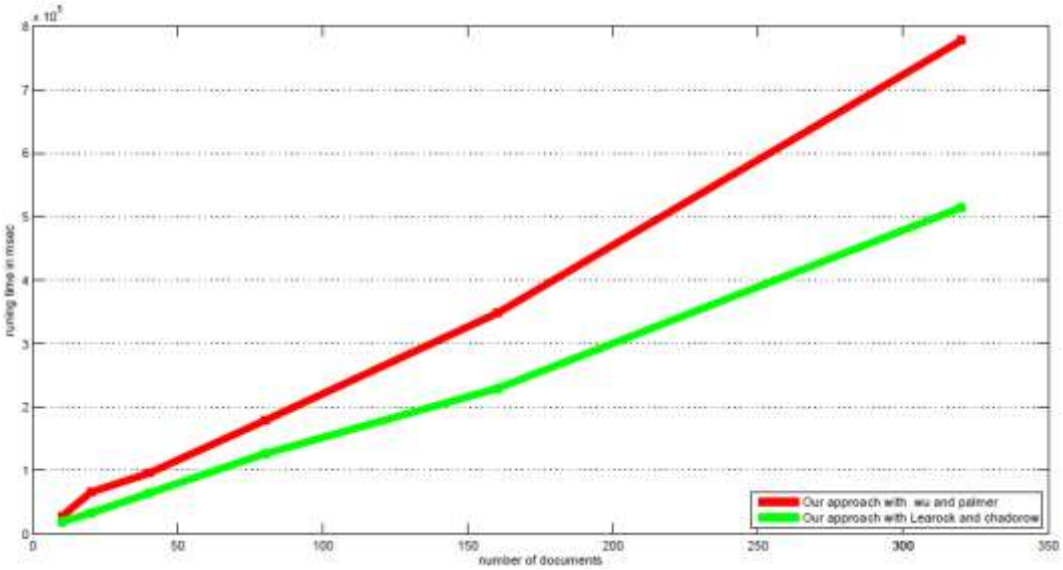

Figure 2. Execution Time using our Approach

Note that for both of our approach, either with Wu and Palmer or with Learock Chodorow if the number of documents increases the time also increases without forgotten

that with the approach of Learock and Chodorow we need an execution time less than that required for the Wu and Palmer approach.

From this experience, we decided to work with the method of Learock and Chodorow in our approach. Figure 3 illustrates the comparison results of our approach with other approaches already existing in the literature, by calculating the semantic similarity between a document and itself.

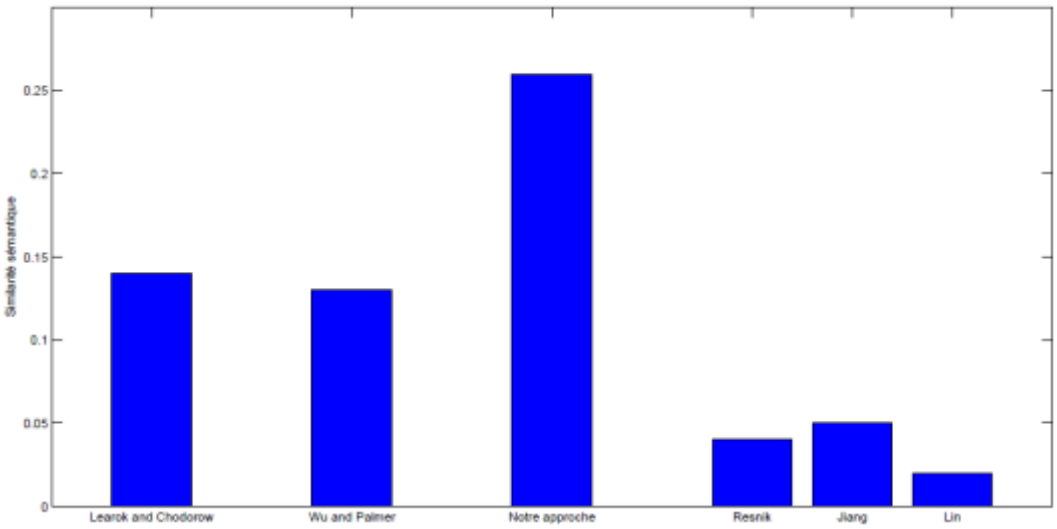

Figure 3. Semantic Similarity for Different Approaches

From this graph, we see that the similarity obtained with our approach outperforms the other approaches. Therefore, we can say that our approach is very effective for an IRS for calculating the semantic similarity between documents.

7. Application of our Approach

The goal of IR is to find the relevant documents to a query, and therefore useful for the user. The quality of a system is measured by comparing the responses of the system with ideal responses that the user expects to receive. More system responses correspond to those that the user hopes better is the system.

In this section, we present the results of the evaluation of our approach to an information retrieval system (IRS), compared with an IRS based on the vector representation and synonymy relations with a matching function based on the approach of cosine.

Comparing the responses of the two IRS, whether it was with our approach or with the approach of cosine (with synonymy) is made by the curve Recall/Precision with two measures, Recall (measures the proportion of relevant documents found among all relevant documents in the base.) and Precision (measuring the proportion of relevant documents found among all the documents found by the system).

Figure 4 shows the steps necessary to calculate the semantic similarity between a query and a set of documents (corpus or collection of documents), like the indexing step that is to say, extract the words of each document and query also the phase of calculating the frequency of each word and the step of using WordNet to find synonymy of words without forgotten the most important phase which is the phase of applying the matching function using our approach.

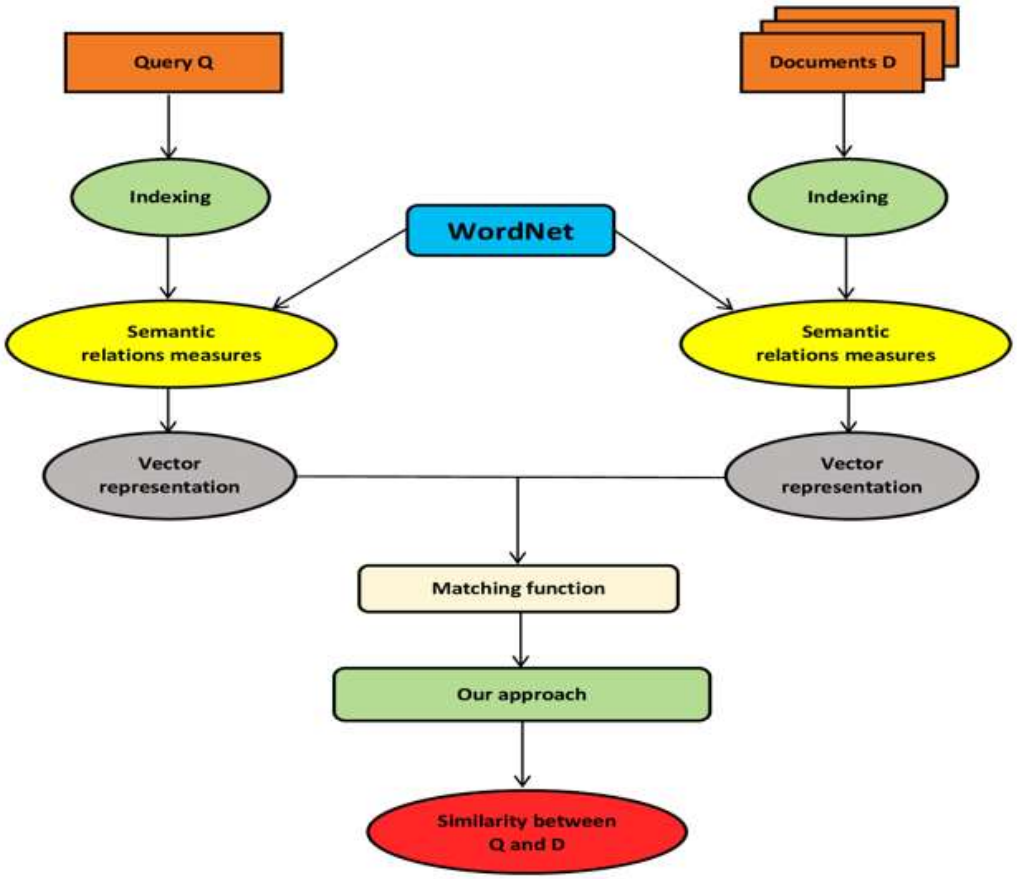

Figure 4. Similarity Calculation Steps using our Approach

We work in this experiment with the results of the past one which showed us that the approach of learock and Chodorow needs a running time less than that required with Wu and Palmer, so we work in this experience with the Learock and Chodorow approach. Figure 5 illustrates the curve recall/precision for both IRS.

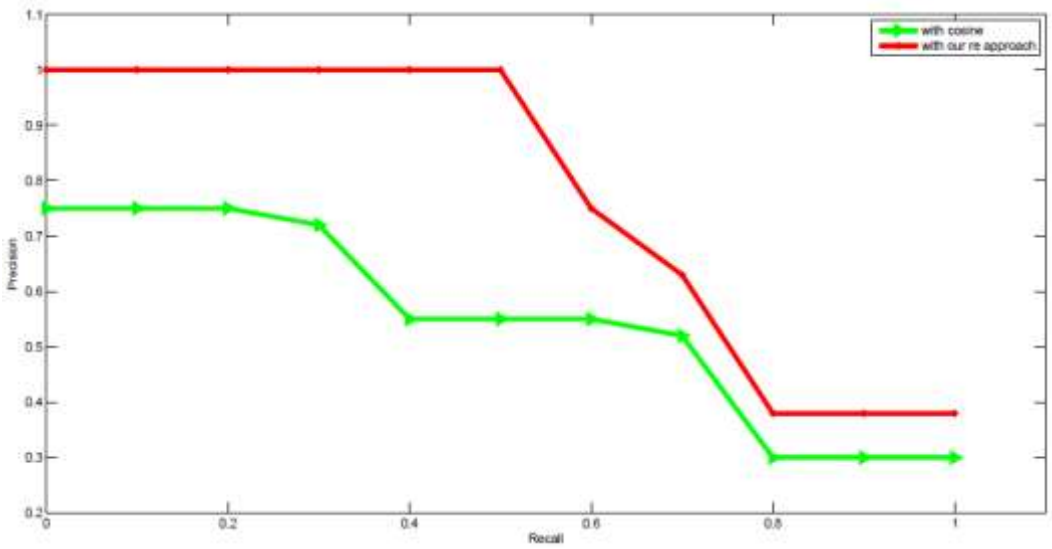

Figure 5. Curve recall/precision for our IRS and IRS of Cosine

Before commenting on this result and as a reminder we can say that if we want to compare two IRS systems they must be tested with the same test corpus. A system whose curve exceeds (that is to say, it is located at the top right of the other) that of the other is regarded as a better system.

So according to this definition and from the curve of Figure 5, we conclude that our IRS is better than the IRS based on Cosine.

8. Big Data Challenges by Hadoop

This is the world of Big Data, Weblogs, Internet texts and Documents, Internet search indexing, Internet of Things (IoT), Cloud Computing…. These all include Big Data for making more strategic decisions and taking the full advantage of available information, it is required to process Big Data efficiently [23].

8.1. Hadoop Clustering

Hadoop is an open source Apache software framework that evaluates gigabytes or petabytes of structured or unstructured data and transforms it more manageable for applications to work with a large data[2].The core components of Hadoop are HDFS and MapReduce. HDFS is basically used to store large data sets and MapReduce is used to process such large datasets.

8.2. Hadoop Distributed File System (HDFS) Architecture

The Hadoop Distributed File System (HDFS) is designed to store very large data sets reliably, and to stream those data sets at high bandwidth to user applications. In a large cluster, thousands of servers both host directly attached storage and execute user application tasks [24]. HDFS uses a write-once, read-many model that breaks data into blocks that it spreads across many nodes for fault tolerance and high performance.

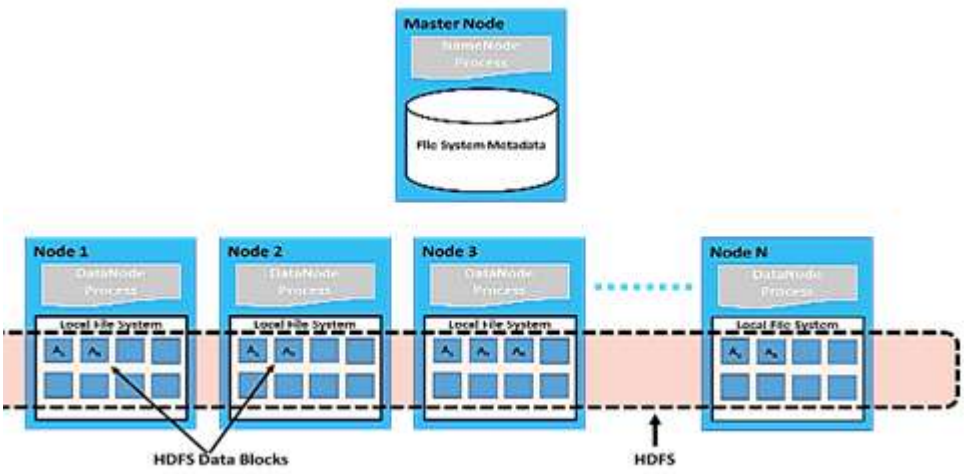

Figure 6. The High-Level Overview of the Structure of HDFS

HDFS stores file system metadata and application data separately on a dedicated server, called the NameNode. Application data are stored on other servers called DataNodes.

NameNode: the node that controls the HDFS. It is responsible for serving any component that needs access to files on the HDFS. It is also responsible for ensuring fault-tolerance on HDFS. Usually, fault-tolerance is achieved by replicating the files over different nodes.

[2] Apache Hadoop, http://hadoop.apache.org/

DataNode: this node is part of HDFS and holds the files that are put on the HDFS. Usually these nodes also work as TaskTracker. JobTracker tries to allocate work to nodes such files accesses are local, as much as possible.

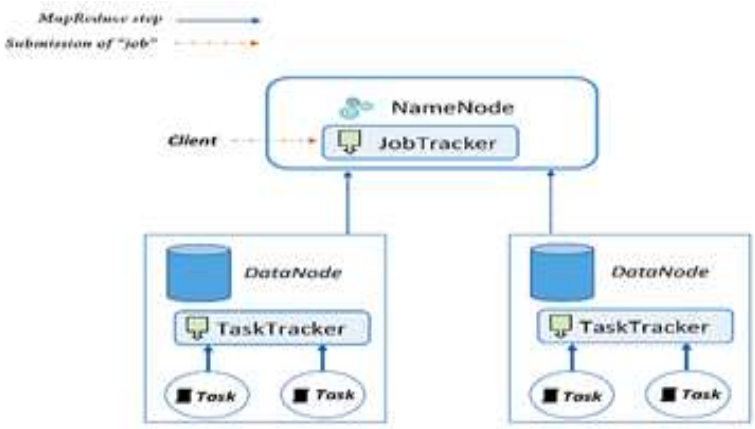

Figure 7. The Interaction between HDFS and MapReduce Job

At the level of the NameNode, the JobTracker is responsible for the management of resources that is the control of the DataNodes in the cluster. It manages the entire duration of the life of a job. The TaskTracker has responsibilities more simple, namely launch the tasks in the order provided by the JobTracker and periodically give a status of progress of the task to the JobTracker.

8.3. MapReduce Programming Model

MapReduce is a programming model and an associated implementation for processing and managing large data sets with a parallel, distributed algorithm on a cluster. MapReduce divides into three parts: Map, Shuffle and Sort, and Reduce. A Map part of MapReduce job splits the input datasets into independent chunks. The independent chunks are processed in a completely parallel manner using Map task. Then the Reduce function merged these values to form a possibly smaller set of values. That is, the Reduce function filtered the Map output and produces the results with respect to the key of the Map phases [25].

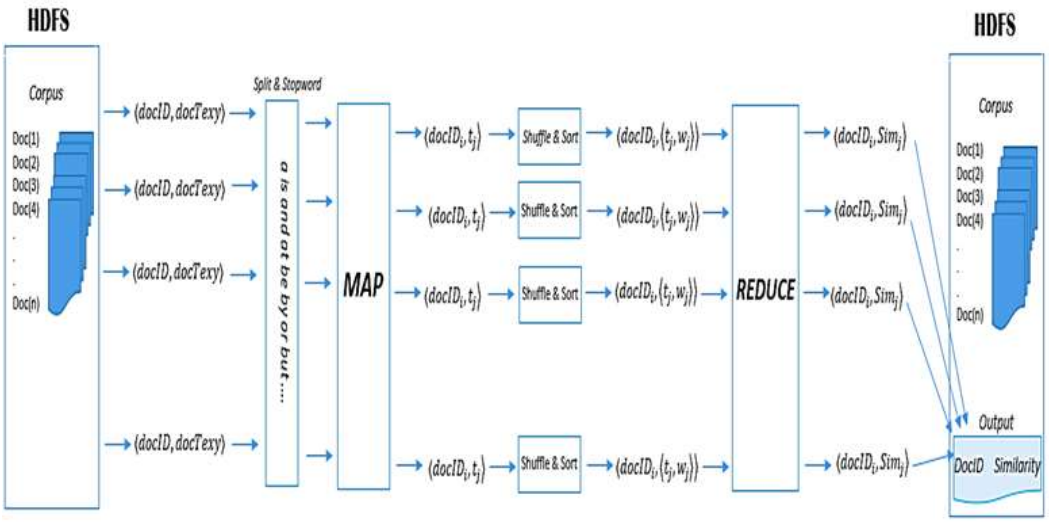

Figure 8. The Process of our MapReduce Model

Our proposed algorithm runs on two consecutive MapReduce phases, the first is to build an indexing phase and the second is to compute the semantic similarity measure, the design of our MapReduce operation is shown in Figure 8.

In our work we propose a new MapReduce algorithm to calculate the semantic similarity between a request in the form of a document, and each document of a corpus choosing in our case that the key for the map operation is the document name (DocID) and the value is the content of each document, so we sent after the map operation a word as value and the name of its document as key, the Reduce phase groups the words of the same document and calculates the semantic similarity using our approach between the request and the document we are working on, at the end of the MapReduce operation we save the result of the calculation of the semantic similarity at the level of HDFS. Each line in HDFS will contain the name of each document and its similarity with the request.

Figure 9 shows the different steps to calculate the semantic similarity using the Hadoop framework.

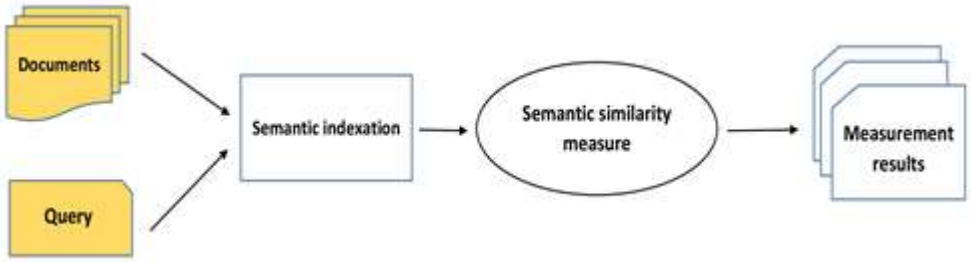

Figure 9. General Scheme of our Work

With:

- Document indexing: given a corpus, for each term of the document, the mapper emits the document ID as the key, and his words as the value. The shuffle phase of MapReduce, groups these words by a collection of the values of each document, and delivers these inverted lists to the reducers, that write them to blocks.
- Semantic similarities measures: In this step, Reduce takes the output of the Map function and computes the semantic relation between each collection of values of each document and the query. This semantic relation computed by WordNet as an external semantic network with the use of the weight of the words and one of the approaches already existed to compute the semantic similarity between the two words.

8.4. Proposed MapReduce Algorithm

The following algorithm shows our MapReduce algorithm for calculating the semantic similarity using our approach.

With:

- CalculateOccurence(e) is a method which calculates the number of occurrences of the word e in the list.
- Indexing(Query) is a method that poster the words exists in the query.


```
Class Mapper
  Method Map(Docid,term)
    RemoveStopWord & removePunctuation (term)
    For each element ∈ (Docid,term)
      Write(Docid,term)
    End for
Class Reducer
  Method Reduce(Docid,List(term))
    List(q) = Indexing(Query)
    S ← 0
    X ← 0
    Y ← 0
    For each n ∈ List(term)
    F=CalculateOccurence(n)
    For each e ∈ List(q)
      R=CalculateOccurence(e)
      X ← X + F * R * Sim(n, e)
      Y ← Y + F * R
    End for
    End for
    S ← X ÷ Y
    Write(Docid,S)
```

9. Experimental Results in Big Data

To evaluate our approach based on the MapReduce programming model and using the HDFS file system, we conduct extensive performance study, the experiment is to run initially our MapReduce algorithm with a variable number of documents in input (in the corpus stored in HDFS), the second experiment is to vary the number of cluster nodes to see the effect of parallelization on the execution time of the MAP and REDUCE functions.

9.1. Result on Documents

Figure 10 illustrates the results obtained with a NameNode and two DataNodes, that is to say, we share the semantic similarity between three machines using the MapReduce programming model.

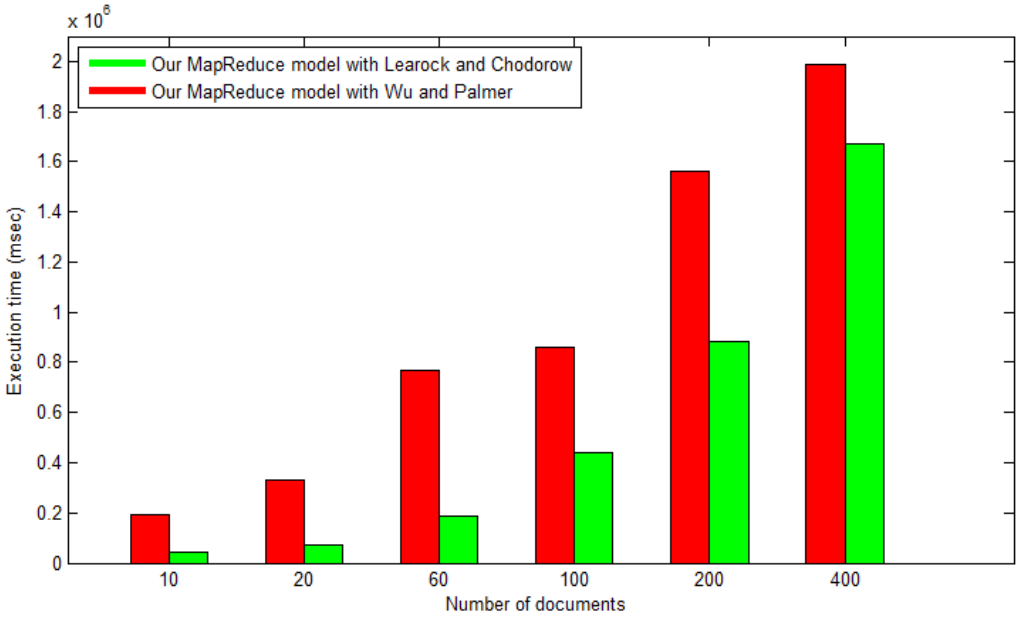

Figure 10. The Execution Time of our Approach with Wu and Palmer and Learock and Chodorow

Therefore, from this figure, we see that when the number of documents increases, the execution time also increases which is normal due to the number of operations in addition whether it was in the Map phase or in the Reduce phase.

Table 4 gives the results of the execution time (msec) on a variable number of documents in the Map and Reduce phase using in our algorithm the Learock and Chodorow approach.

Table 4. Evaluation of the Execution Time of our Approach in Hadoop

Number of documents	Map	Reduce
10	2974	40636
20	4845	75791
60	16381	172799
100	32562	406338
200	68891	812766
400	100336	1568975

9.2. Results on the Nodes

Figure 11 gives the results obtained for the calculation of the semantic similarity between a query and a document collection, using a NameNode and a variable number of DataNodes, each time comparing the execution time of the Map and Reduce phase.

Note that if the number of nodes increases, the running time in the stage Map and Reduce decrease. Which is the advantage of our algorithm and the objective of working in a distributed system that is to say reducing the time of calculation of the semantic similarity although the corpus contains a large number of documents.

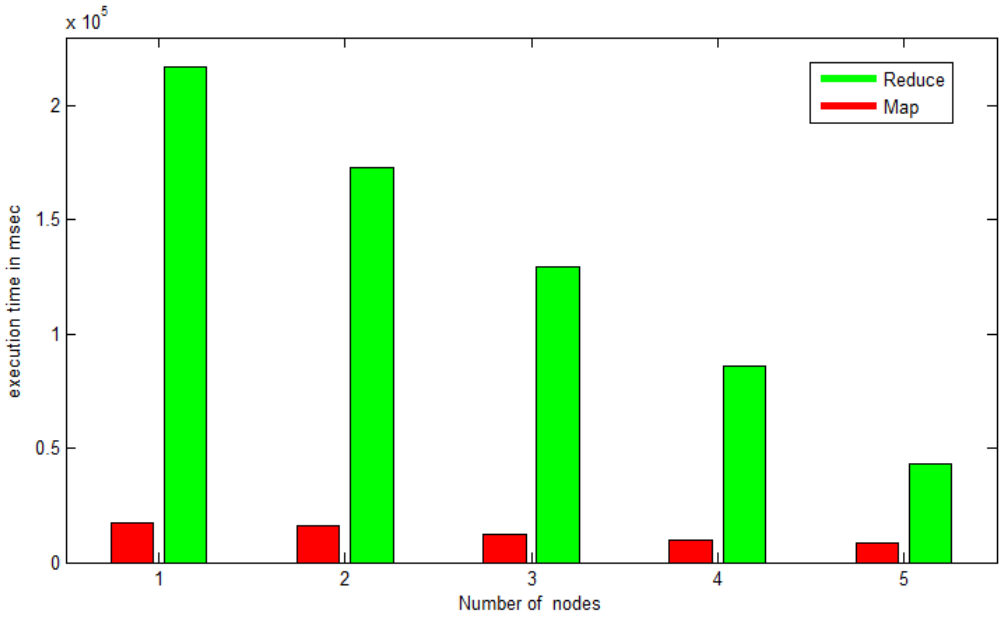

Figure 11. The Execution Time of the Map and Reduce by Varying the Number of Nodes

10. Conclusion and Future Works

In this work, we presented an extension of similarity measure based on two representation methods, the vector representation using semantic relationship which is the synonymy, and the second representation is to use existing approaches in the literature. We have shown that the use of synonymy added positive value to the proposed approach. We compare our approach with existing measures in the literature like Wu and Palmer or Learock and Chodorow. The Experimental results show that the proposed approach despite it requires a relatively large execution time than other approaches but it gives a great precision (Relevance) than other approaches when calculating the semantic similarity between two documents.

The problem we found when calculating the semantic similarity between documents in a single system with a single server is the time we need to wait to see the result, but with the Big Data we exceeded this problem by working with the distributed system (sharing the work between several machines).

10.1. Future Works

- the application of our approach to another language like Arabic to make an Arabic IRS.
- Reduce the execution time of the Reduce operation at the proposed MapReduce algorithm, because as we saw in the experimental results, the Reduce stage needs a relatively long time to do its work, our proposal to work with multiple MapReduce that is to say uses the result of the first MapReduce to run another.

References

[1] F. Harrathi, "Extraction de concepts et de relations entre concepts à partir des documents multilingues : Approche statistique et ontologique", L'Institut Nationale des Sciences Appliquées de Lyon, **(2009)**.
[2] R. Harrathi, "Recherche d'information conceptuelle dans les documents semi-structurés", L'Institut Nationale des Sciences Appliquées de Lyon, **(2010)**.
[3] D. Lin. "An Information-Theoretic Definition of similarity", In Proceedings of the Fifteenth International Conference on Machine Learning (ICML'98). Morgan-Kaufmann: Madison, WI, **(1998)**.

[4] P. Resnik. "Semantic similarity in a taxonomy: An information based measure and its application to problems of ambiguity in natural language", Journal of Artificial Intelligence Research, vol. 11, **(1999)**, pp. 95-130.
[5] R. Rada, H. Mili, E. Bichnell and M. Blettner, "Development and application of a metric on semantic nets", IEEE Transaction on Systems, Man, and Cybernetics, **(1989)**, pp 17-30.
[6] Z. Wu and M. Palmer, "Verb semantics and lexical selection", In Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics, **(1994)**, pp 133-138.
[7] J. Jiang and D. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy", In Proceedings of International Conference on Research in Computational Linguistics, Taiwan, **(1997)**.
[8] C. Leacock and M. Chodorow, "Combining Local Context and WordNet Similarity for Word Sense Identification", In WordNet: An Electronic Lexical Database, C. Fellbaum, MIT Press, **(1998)**.
[9] G. Salton and M. J. McGill, "Introduction to modern information retrieval", McGraw-Hill. New York, **(1983)**.
[10] A. Chowdhury, "Duplicate data detection", retrived from http://ir.iit.edu/~abdur/Research/Duplicate.html, **(2004)**.
[11] C. Lyon, J. Malcolm and B. Dickerson, "Detecting Short Passages of Similar Text in Large Document Collections", Processing of EMNLP, **(2001)**.
[12] I. Matveeva, "Document Representation and Multilevel Measures of Document Similarity", Proceedings of ACLHLT, **(2006)**.
[13] Q. Zhang, Y. Zhang, H. Yu and X. Huang, "Efficient Partial-Duplicate Detection Based on Sequence Matching", Proceedings of SIGIR, **(2010)**.
[14] P. Siddharth, S. Banerjee and T. Pedersen, "Using measures of semantic relatedness for word sense disambiguation", In Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, Mexico, **(2003)**, pp. 241-257.
[15] M. Diana, R. Koeling, J. Weeds and J. Carroll, "Finding predominant senses in untagged text", In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Barcelona, Spain, **(2004)**, pp. 280-287.
[16] I. Gurevych and M. Strube, "Semantic similarity applied to spoken dialogue summarization", In Proceedings of the 20th International Conference on Computational Linguistics, Geneva, Switzerland, vol. 23-27, **(2004),** pp. 764-770.
[17] G. Hirst and A. Budanitsky, "Correcting real-word spelling errors by restoring lexical cohesion", Natural Language Engineering, **(2004)**.
[18] Y. Madani, M. Erritali and J. Bengourram, "Arabic Stemmer Based Big Data", J. Electron. Commer. Organ. JECO, vol. 16, no 1, **(2018)** January, pp. 17-28.
[19] P. W. Lord, R. D. Stevens, A. Brass and C. A. Goble, "Semantic Similarity Measures as Tools for Exploring the Gene Ontology", Pacific Symposium on Biocomputing, vol. 8, **(2003)**, pp. 601-612.
[20] H. Jeffrey, L. William and D. John, "A Semantic Similarity Measure for Semantic Web Services", In proceedings of WSS05, **(2005)**.
[21] F. Chen, A. Farahat and T. Brants, "Multiple Similarity Measures and Source-Pair Information in Story Link Detection", In Proceedings of HLT, **(2004)**.
[22] G. A. Miller, "WordNet: A Lexical Database for English", Communications of the ACM, vol. 38, no. 11, **(1995)**, pp. 39-41.
[23] H. Bagheri and A. Abdullah Shaltooki, "Big Data: Challenges, Opportunities and Cloud Based Solutions", International Journal of Electrical and Computer Engineering (IJECE), vol. 5, no. 2, **(2015)** April, pp. 340-343.
[24] K. Shvachko, H. Kuang, S. Radia and R. Chansler, "The Hadoop Distributed File System".
[25] Y. B. Reddy, "Document Identification with MapReduce Framework", Data Analytics: The Third International Conference on Data Analytics, **(2014)**.

