Indonesian Journal of Electrical Engineering and Computer Science
Vol. 5, No. 1, January 2017, pp. 196~ 205
DOI: 10.11591/ijeecs.v5.i1.pp 196-205

Ontology Matching Using BabelNet Dictionary and Word Sense Disambiguation Algorithms

Mohamed Biniz*, Rachid el Ayachi, Mohamed Fakir
Laboratory of Information Processing and Decision Support (TIAD), Faculty of Sciences and Technics, University Sultan Moulay Slimane Beni Mellal, Morocco
Corresponding author, e-mail: mohamedbiniz@gmail.com*, rachid.elayachi@usms.ma, fakfad@yahoo.fr

Abstract

Ontology matching is a discipline that means two things: first, the process of discovering correspondences between two different ontologies, and second is the result of this process, that is to say the expression of correspondences. This discipline is a crucial task to solve problems merging and evolving of heterogeneous ontologies in applications of the Semantic Web. This domain imposes several challenges, among them, the selection of appropriate similarity measures to discover the correspondences. In this article, we are interested to study algorithms that calculate the semantic similarity by using Adapted Lesk algorithm, Wu & Palmer Algorithm, Resnik Algorithm, Leacock and Chodorow Algorithm, and similarity flooding between two ontologies and BabelNet as reference ontology, we implement them, and compared experimentally. Overall, the most effective methods are Wu & Palmer and Adapted Lesk, which is widely used for Word Sense Disambiguation (WSD) in the field of Automatic Natural Language Processing (NLP).

Keyword: *matching ontology, web ontology language, semantic web, semantic similarity, word sense disambiguation, natural language processing, ontology, babelnet similarity*

1. Introduction

The last ten years have seen advances in information and communications technology who provided a huge amount of heterogeneous information. This development has been driven by the standardization of knowledge representation languages on the semantic web, including Resource Description Framework (RDF) and Web Ontology Language (OWL) [1]. In practice, the developed ontologies are constructed independently of each other by different organizations. This causes the problem of heterogeneity changes direction or ambiguity in the interpretation of the entities, and therefore, it prevents the sharing domain knowledge. It becomes essential to establish and to integrate semantic correspondences between ontologies. This integration is based on a task called alignment whose role is to determine the best matches between the data source elements. Difficulties are first to maintain a good quality of the match, on the other hand, to ensure acceptable performance with regard to the size and number of processed sources.

Today, many techniques have been developed, or borrowed from other areas to discover semantic mappings between entities. Among these methods, terminological matching is a very popular comparison class names and property of ontologies using a string distance metric to produce a degree of similarity.

We propose in this paper an alignment approach, which combines different techniques from the field of automatic processing of natural language (Adapted Lesk algorithm, the Wu & Palmer algorithm, Resnik algorithm Leacock and Chodorow algorithm), to design an efficient similarity measure for comparing ontology entities. This paper is organized as follows: Section 2 discusses the background and purpose of the algorithms used in this work. Section 3 deals with the architecture of the proposed system. Experiments results and discussion are given in section 4.

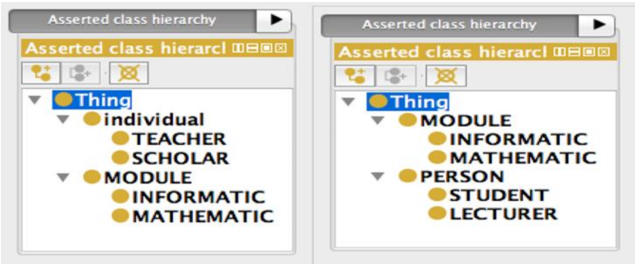
Received September 28, 2016; Revised December 2, 2016; Accepted December 19, 2016

2. Background and Purpose of Work

The similarity measures have been used in text classification [2], question analysis [3, 4] and word Sens disambiguation [5], etc.

In classical approach [6], to find the similarity between two documents is to use a simple syntactic method, and calculate a similarity score based on the number of tokens that occur in the two documents. The improvements have been made to this simple method by removing stop words and considering only the longest subsequence, or weighting and normalization. For example, it is difficult to find a strong similarity between I filed a bag of wheat in my account and I shed $100,000 in my bank. In the context of our study, several words can be used to talk about the same subject. Therefore, take into account the semantic seems very important.

Figure 1 describes two simple ontologies modeled from two universities for the same requirement. Person is represented in this university by two classes (Lecturer and students). The university offers a number of computer modules, and Mathematics. Each lecturer teaches one or more modules and each student can study one or more modules.



Ontology O_1 Ontology O_2

Figure 1. O_1 and O_2 are two ontologies for two universities

Both Teacher and Lecturer classes are synonyms for the entity Professor, but there is no correspondence between these two classes using a threshold of 0.7, because the similarity with $\text{Levenshtein}(\text{Lecturer}, \text{Teacher}) = 0.65$ or $\text{Jaccard}\ (\text{Lecturer}, \text{Teacher}) = 0.5$,

2.1. BabelNet

BabelNet [7] is the largest multilingual encyclopedic dictionary and semantic network created by means of the seamless integration of the largest multilingual Web encyclopedia.

2.2. Word Sense Disambiguation Algorithms

2.2.1. Adapted Lesk and Lesk Algorithms

Lesk [8] proposed a very simple word sense disambiguation algorithm, which considers the similarity between two senses as the number of words in common in their definitions. In the original version, it does not take into account the order of words in the definitions (bag of words). The similarity of Lesk is expressed by equation (1).

$$\text{LESK}(\text{CLASS1}, \text{CLASS2}) = \text{DEFINITION}(\text{CLASS1}) \cap \text{DEFINITION}(\text{CLASS2}) \tag{1}$$

Pederson and Banerjee [9] proposed an improved Lesk, called Adapted Lesk and defined by equation (3), based on two steps. The first step is the incorporation of the definitions of sense connected by BabelNet taxonomic relationships in the definition of a given sense. In the second step, it calculates the overlap between the definitions of words by considering not only the overlap between the definitions of the two senses but also the definitions of relations R: hyperonyms (has-kind), hyponyms (kind-of) meronyms (part-of) holonyms (has-hand), but also by troponymes attribute relations (similar-to and also-see). To ensure that the measure is symmetric, the authors propose to group the recovery assessments between the definitions of pairs relationships $\mathfrak{R}$.

Let ψ be the series of connections to calculate the recovery. A set is defined by equation (2):

$$\Re = \{(R1, R2) | \forall (R1, R2) \in \psi^2, (R1, R2) \in \Re^2 \Rightarrow (R1, R2) \in \Re^2\} \quad (2)$$

Therefore, the score is calculated as the sum of overlap between the definitions of pairs of relationships:

$$\mathrm{ADLESK}(C1, C2) = \sum_{\forall (R1,R2) \in \Re^2} (|\mathrm{DEFINITION}(R1(C1)) \cap \mathrm{DEFINITION}(R2(C2))|)^2 \quad (3)$$

2.2.2. Wu & Palmer Algorithm

The similarity of Wu and Palmer[10]between two classes is calculated by using equation (4).

$$\mathrm{WUP}(Class\ 1, \mathrm{Class2}) = \frac{2 * \mathrm{depth(LCS)}}{\mathrm{depth(Class1)} + \mathrm{depth(CLASS2)}} \quad (4)$$

Where

1. depth(LCS): is the depth of the lowest common ancestor between two classes and the root.
2. depth(Class1) is the depth from the root to class1
3. depth(Class2) is the depth from the root to class2

Figure 2 representsthe taxonomy of university.

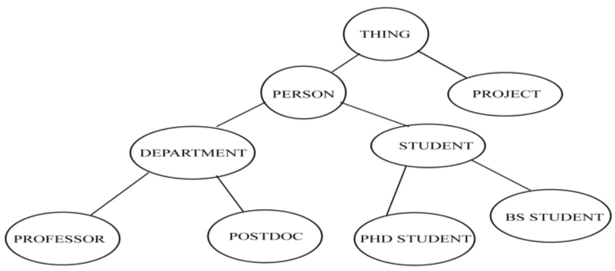


Figure 2. Ontology of a university

From the taxonomy of Figure 2, if we consider the concepts PROFESSOR, BS-STUDENT and POSTDOC, knowing that the LCS of PROFESSOR and BS-STUDENT is PERSON and the LCS of PROFESSOR and POSTDOC is DEPARTMENT, we have:

$$\mathrm{WUP(PROFESSOR, BS\ STUDENT)} = \frac{2 * \mathrm{depth(PERSON)}}{\mathrm{depth(PROFESSOR)} + \mathrm{depth(BS\ STUDENT)}} = \frac{2 * 1}{3 + 3} = 0.33$$

and

$$\mathrm{WUP(PROFESSOR, POSTDOC)} = \frac{2 * \mathrm{depth(DEPARTMENT)}}{\mathrm{depth(PROFESSOR)} + \mathrm{depth(POSTDOC)}} = \frac{2 * 2}{3 + 3} = 0.66$$

Consequently, PROFESSOR and POSTDOC are closer to each other than BS-STUDENT.

2.2.3. Leacock and Chodorow Algorithm

Leacock and Chodorow [11] used a single relation (hyponymy) and changed the formula of path length to reflect the fact that the lower arch in the hierarchy hyponymy corresponds to the smallest semantic distance. The similarity between the two concepts Class1 and Class2 is given by equation (5):

$$\mathrm{LCH(CLASS1, CLASS2)} = -\log\left(\frac{\text{shorter path length(CLASS1, CLASS2)}}{2 * \text{depth}}\right) \quad (5)$$

From the taxonomy of Figure 2, if we consider the concepts PROFESSOR, POSTDOC and BS-STUDENT, we have:

$$\mathrm{LCH(PROFESSOR, BS\ STUDENT)} = -\log\left(\frac{4}{2 * 3}\right) = 0.176$$

and

$$\mathrm{LCH(PROFESSOR, POSTDOC)} = -\log\left(\frac{2}{2 * 3}\right) = 0.4771$$

Consequently, PROFESSOR and POSTDOC are closer to each other than BS-STUDENT.

2.2.4. Resnik Algorithm

The resnik measure [12] returns the Information Content (IC) of the Lowest Common Subsumer (LCS) of two given concepts, it is calculated by equation (6):

$$\mathrm{RES(CLASS1, CLASS2)} = \mathrm{IC}\big(\mathrm{LCS(CLASS1, CLASS2)}\big) = -\log\Big(\mathrm{P}\big(\mathrm{LCS(CLASS1, CLASS2)}\big)\Big) \quad (6)$$

From the taxonomy of Figure 2, if we consider the concepts PROFESSOR, BS-STUDENT and POSTDOC, knowing that the LCS of PROFESSOR and BS-STUDENT is PERSON and the LCS of PROFESSOR and POSTDOC is DEPARTMENT, and if we assume that IC(PROFESSOR)= IC(BS STUDENT) = 0.4, IC(POSTDOC) = 0.5, IC(PERSON) = 0.3 and IC (DEPARTMENT) = 0.5, we have:

$$\mathrm{RES(PROFESSOR, BS\ STUDENT)} = \mathrm{IC}\big(\mathrm{LCS(PROFESSOR, BS\ STUDENT)}\big) = \mathrm{IC(PERSON)} = 0.3$$

and

$$\mathrm{RES(PROFESSOR, POSTDOC\ )} = \mathrm{IC}\big(\mathrm{LCS(PROFESSOR, POSTDOC\ )}\big) = \mathrm{IC(DEPARTMENT)} = 0.5$$

2.3. Similarity Flooding Algorithms

The Similarity Flooding [13] Algorithm (SFA) takes two graphs as input, and produces as output a mapping between corresponding nodes of the graphs.

1. As a first step, we transform ontologies in a graph G in which the vertices are pairs of ontology concepts and edges exist between two nodes, if there is a relationship in both ontologies between the nodes of the two pairs. In fact, the original similarity flooding algorithm only connects the concepts whose edges have the same label.
2. As a second step, we assign weight w to the edges, which are typically 1/n, where n is the number of outgoing edges.
3. As a third step, we assign initial similarity σ^0 to each node.
4. As a fourth step, we compute $\sigma^{i+1}$for each node with the following formula (7):

$$\sigma^{i+1}(x, x') = \sigma^i(x, x') + \sum_{(\langle y, y'\rangle, p, \langle x, x'\rangle)} \sigma^i(y, y') \times w(\langle y, y'\rangle, p, \langle x, x'\rangle) \quad (7)$$

Where:

1. $\sigma^{i+1}(x, x')$ is the similarity value between two entities (x, x') in the iteration *i+1.*
2. $\sigma^i(x, x')$ is the similarity value between two entities (x, x') in the iteration *i.*
3. $\sigma^i(y, y')$ is the similarity value between two entities (y, y') in the iteration *i.*
4. $w(\langle y, y'\rangle, p, \langle x, x'\rangle)$ is the weight of the outgoing arc from entities (y, y') to entities in (x, x').

5. As a fifth step, we normalize all $\sigma^{i+1}$by dividing by the largest value.
6. As a sixth step, if no similarity changes more than threshold ϵ, or after prefixed number of steps, stop otherwise go to the fourth step.

2.4. Existing Matching Tools

2.4.1. Shiva and Shiva++

Shiva [14] and Shiva++ [15] are a semi-automated ontology alignment system. They were designed to allow the discovery of correspondence between two ontologies. The system manages ontologies specified in Web Ontology Language (OWL), Resource Description Framework (RDF) or Extensible Markup Language (XML) formats. Shiva and Shiva++ inputs two ontologies and outputs (one-to-one or one-to-many) correspondences between concepts of these ontologies inOntology Alignment Evaluation Initiative (OAEI) format. The approach is based on the construction of a matrix of a bipartite graph, its vertices are the concepts and its edges are the values of the similarity between the concepts through algorithms similarities such as Edit Distance, Qgrams, Smith Watermanand jaccard's coefficient algorithms. All the algorithms search for similarities between concepts, sub-concepts, properties and instances. Once the matrix is generated, it is passed by a graph matching algorithm called the Hungarian method [16] to extract the correspondences.

2.4.2. FOAM

Framework for Ontology Alignment and Mapping (FOAM) [17] is a system that fully or semi-automatically align two or more OWL Ontologies, based on heuristics (similarity) of the individual entities (concepts, relations, and instances). it includes six essential steps:

1. Feature engineering selects extracts from the general definition of ontology to describe a specific entity.
2. Search step selection chooses two entities from the two ontologies to compare (E_1, E_2).
3. Similarity assessment indicates a similarity for a given description of two entities.
4. Similarity aggregation aggregates the multiple similarity assessments for one pair of entities into a single measure.
5. Interpretation uses all aggregated numbers, a threshold and an interpretation strategy to propose the alignment.
6. Iteration is based on the similarity of one alignment, this similarity is influenced by the neighboring similarity between every entity pairs; the equality is propagated through the ontologies.

3. Architecture of the Proposed System

The goal of our approach is to provide a model of integration, flexible and powerful at the same time, able to unite the different heterogeneous data sources. For this, the definition of our architecture must include a minimum number of steps to handle.

As we can see in Figure 3, our approach is composed of three distinct stages, namely:

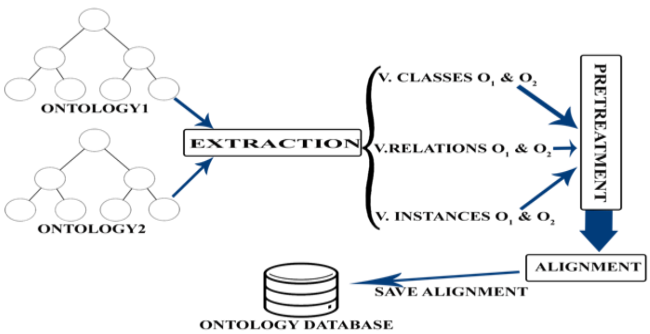


Figure 3. The alignment steps of two ontologies O_1 and O_2

1. The extraction step: this step allows to build three vectors for each ontology, the first contains classes. The second consists of the relationships of each ontology and the third contains the individuals.
2. The pretreatment step: takes as input the six vectors (three vectors for each ontology) of the previous step. This step has the role of standardizing the elements of these vectors, for example remove the spaces, put all the elements in lowercase, and detecting the part of speech of each element ([teacher: name]; [learn: verb], …etc.).
3. The alignment step: for each type of vector (example vector class of O_1 and O_2) calculates the semantic similarity between the two elements of the vector using the algorithms explained previously and BabelNet.

Figure 4 shows an example of alignment between elements of two vector classes using a threshold 0.8. In step align, we use algorithms of word sense disambiguation. Therefore, this step gives the degree of correspondence between two elements, one of the first vectors and the other of the second.

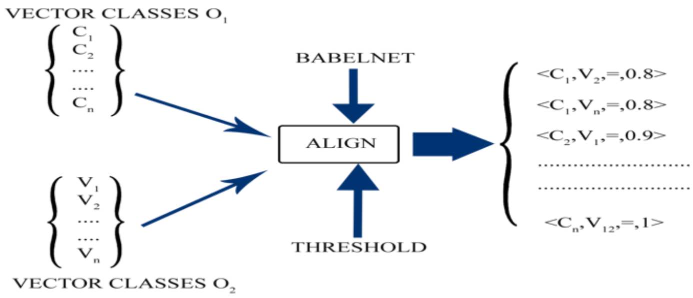


Figure 4. Alignment of the two vectors elements

To demonstrate the performance of the approach we used the reference tests OAEI 2014 [18]. The domain of this test is bibliographic references. It is, of course, based on a subjective view of what must be a bibliographic ontology. There can be many different classifications of publications (based on the area, quality, etc.). We choose the one common among scholars based on means of publications.

To evaluate the experiments results a definition of evaluation metrics is required. The metrics for measuring performance are theprecision, recall and F-measure [19].

$$\text{precision} = \frac{\text{good pair align found}}{\text{pair align found}} \quad (8)$$

$$\text{recall} = \frac{\text{good pair align found}}{\text{pair align existing}} \quad (9)$$

$$\text{F} - \text{measure} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}} \quad (10)$$

4. Experimental Analysis

Tables 1, 2, 3, and 4 represent respectively the recall, precision and F-measure obtained by our system throughout the process. We see that the alignment that is based on the algorithm or Leacok Resnik & Chodorow algorithm produces a low recall and precision, we find that these two algorithms are effective to find as many good matches, but also generate many false matches. For the alignment with Wu & Palmer and Lesk adapted algorithms,we obtain a high level of accuracy. For example, Wu & Palmer metric achieves better recall in experiences, but the precision is so poor between 10 and 30 pairs of false negatives, to be removed later, perhaps by human intervention or use of a richer dictionary.

The threshold is 0.8. Indeed, those algorithms have more similarities than the predefined threshold. For tests from 248 to 266, the system gives bad results since it does not take into account the lexical and structural difference.

Table 1. Alignment results using Adapted Lesk Algorithm

test	precision	recall.	F-measure
101 to 104	0.72	0.64	0.67
201 to 210	0.43	0.52	0.47
221 to 247	0.67	0.66	0.66
248 to 266	0.01	0.2	0.01
301 to 304	0.78	0.82	0.79

Table 2. Alignment results using Wu & Palmer Algorithm

test	precision	recall.	F-measure
101 to 104	0.82	0.84	0.82
201 to 210	0.54	0.62	0.57
221 to 247	0.77	0.71	0.73
248 to 266	0.10	0.10	0.10
301 to 304	0.8	0.84	0.81

Table 3. Alignment results using Leacok & Chodorow Algorithm

test	precision	recall.	F-measure
101 to 104	0.43	0.5	0.46
201 to 210	0.32	0.42	0.36
221 to 247	0.65	0.41	0.50
248 to 266	0.00	0.00	0.00
301 to 304	0.43	0.5	0.46

Table 4. Alignment results using Resnik Algorithm

test	precision	recall.	F-measure
101 to 104	0.4	0.34	0.36
201 to 210	0.31	0.2	0.24
221 to 247	0.45	0.31	0.36
248 to 266	0.00	0.00	0.00
301 to 304	0.44	0.63	0.51

However, we correct this problem by the use of similarity flooding algorithm. So,we compare our approach with other ontology alignment systems on test data sets. Figures 5 and 6 show the results of the alignment systems based on their precision (Figure 5) and their recall (Figure 6) on the dataset of Benchmark Library 2014.

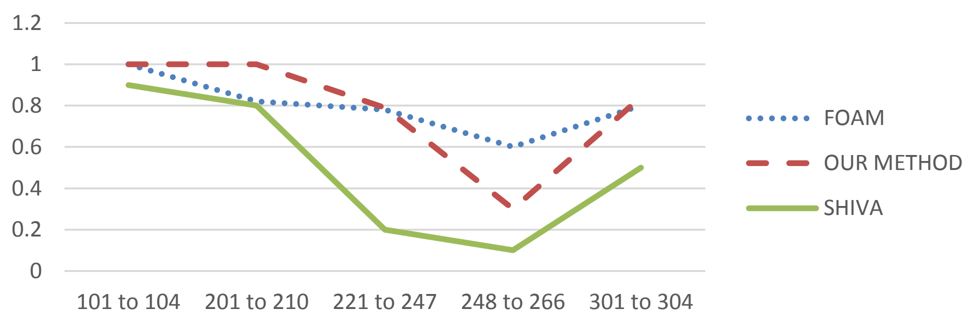


Figure 5. Results of the alignment systems based on the precision

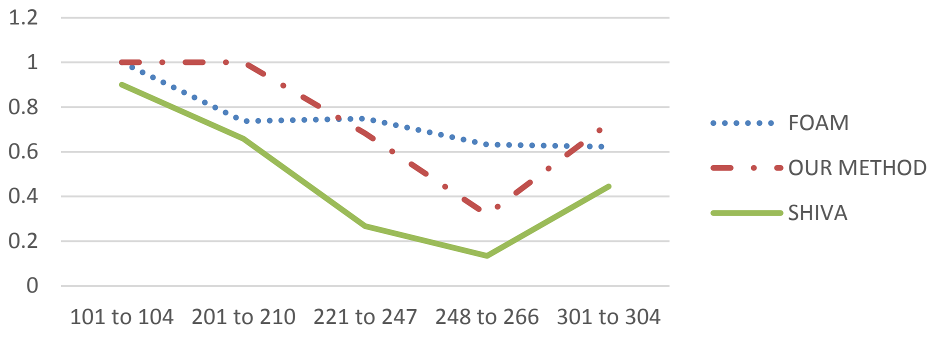


Figure 6. Results of the alignment systems based on therecall

As seen in Figure 5 and 6, our approach outperforms systems studied, except FOAM system in the tests from 248 to 266, for which the precision is 0.6 (recall is 0.67) and for our system accuracy is 0.3 (recall is 0.34). This difference is due to the nature of the bases from 248 to 266 tests (these bases have no label or comment or identifier that return zero results for the calculation of similarity between the entities), which generates mismatch correspondences. Unlike FOAM, our approach uses external resources such as Babelnet requests in version 3.7.

In Figure 7, the performance of all systems are measured with F-measure. Our system outperforms systems are studying with 0.86 average F-measure. However, Shiva depends heavily lexical information and uses only some specific relationship relations. So when the ontologies to align contain many other relationships, it is difficult to take into account the structural information in the alignments. In fact, it is one of the worst systems for all of the benchmark tests.

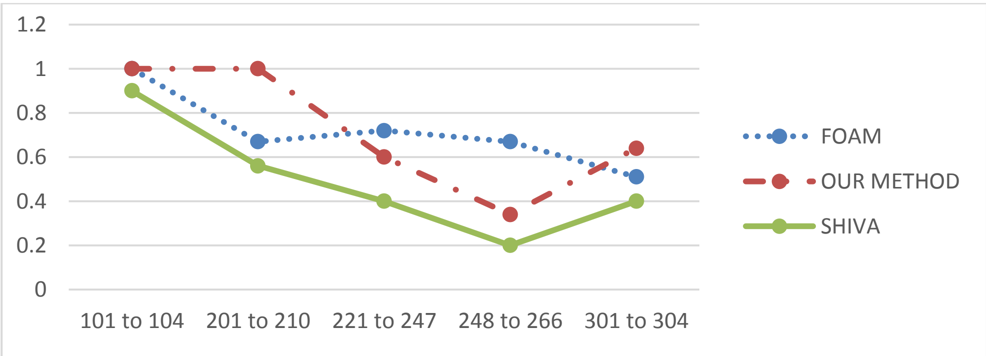


Figure 7. Results of the alignment systems based on the F-measure

For ontologies numbered from 201 to 210: Shiva and FOAM systems give poor performance in alignment with these ontologies because they only use lexical information. An interesting result is that our system gives good results.

Ontologies numbered from 221 to 247 are designed to show the ability of each system to deal with the structural difference between ontologies. Therefore, only systems using structural information should achieve good performance. However, our approach and system FOAM reach good results.

For ontologies from 248 to 266 are the most difficult to align, as they are generated by modifying both lexical and structural information of the reference ontology. Thus, systems must take into account the lexical and structural difference of ontologies at the same time for the alignment of these ontologies. However, most systems consider that one of them or treat lexical information first. As a result, their performance is extremely low when the lexical information is not available.

For real test bases (from 301 to 304), our approach gives better results over systems are studying because it takes into account the linguistic information (synonymous).

Experiments show that our system works well. Nevertheless, the number of false matches is still very high when only the linguistic information, hence the usefulness of combining structural and linguistic measures to reduce the number of false matches, that is to say, improving recall without deteriorating the precision or increase them.

5. Conclusion

In this article, we presented a system to calculate the similarity of two entities (classes, relationships, … etc.) of two different ontologies. These ontologies are represented in RDF graphs. The measure of correspondence is based on BabelNet and disambiguation Algorithms. We gave a definition of BabelNet, and we have proposed algorithms to calculate the similarity of two sets. The system has been implemented in Java and OWL API [20].

To evaluate the performance of our system, it was tested with entities in OWL ontologies benchmark OAEI 2014 and compared with other systems. Indeed, we have achieved significant results for the alignment that uses the linguistic and structural information.

References

[1] PR Swaminarayan. Knowledge representation in Semantic Web and Development of Academic Ontology using Web Ontology Language & SWRL. *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*. 2014; 3(1).

[2] E Moreau, F Yvon and O Cappé. Appariement d'entités nommées coréférentes: combinaisons de mesures de similarité par apprentissage supervisé, in *TALN 2008*, 2008, p. Online–Proceedings.

[3] AE Karyawati, E Winarko, A Azhari and A Harjoko. Ontology-based Why-Question Analysis Using Lexico-Syntactic Patterns. *International Journal of Electrical and Computer Engineering*. 2015; 5(2): 318.

[4] J Yu, D Li, S Hao, J Hou and J Wang. Applying Ontology and VSM for Similarity Measure of Test Questions. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2014; 12(9).

[5] DSJG Tchechmedjiev. Désambiguïsation lexicale de textes : efficacité qualitative et temporelle d'un algorithme à colonies de fourmis. 2013.

[6] W.Cohen, P Ravikumar and S Fienberg. A comparison of string metrics for matching names and records. *Kdd workshop on data cleaning and object consolidation*. 2003; 3: 73–78.

[7] R Navigli and SP Ponzetto. BabelNet: Building a very large multilingual semantic network. *Proceedings of the 48th annual meeting of the association for computational linguistics*. 2010: 216–225.

[8] M Lesk. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. in *Proceedings of the 5th Annual International Conference on Systems Documentation*, New York, NY, USA. 1986: 24–26.

[9] S Banerjee. Adapting the Lesk algorithm for word sense disambiguation to WordNet. University of Minnesota Duluth. 2002.

[10] Z Wu and M Palmer. Verbs semantics and lexical selection. in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. 1994: 133–138.

[11] A Budanitsky and G Hirst. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*. 2006; 32(1): 13–47.

[12] P Resnik and others. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *J. Artif. Intell. Res. (JAIR)*. 1999; 11: 95–130.

[13] S Melnik, H Garcia-Molina and E Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. *Proceedings. 18th International Conference on Data Engineering, 2002*. 2002: 117–128.

[14] I Mathur, N Joshi, H Darbari and A Kumar. Shiva: A Framework for Graph Based Ontology Matching. *International Journal of Computer Applications*. 2014; 89(11).

[15] Iti Mathur, N Joshi, H Darbari and A Kumar. Shiva++: An Enhanced Graph based Ontology Matcher. *International Journal of Computer Applications*. 2014; 92(16).

[16] ML Balinski and RE Gomory. A primal method for the assignment and transportation problems. *Management Science*. 1964; 10(3): 578–593.

[17] M Ehrig and Y Sure. Foam-framework for ontology alignment and mapping results of the ontology alignment evaluation initiative. *Integrating Ontologies Workshop Proceedings*. 2005; 72.

[18] J Euzenat, ME Roşoiu and C Trojahn. Ontology matching benchmarks: Generation, stability, and discriminability. *Web Semantics: Science, Services and Agents on the World Wide Web*. 2013; 21: 30–48.
[19] DM Powers. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. 2011.
[20] M Horridge and S Bechhofer. The owl api: A java api for owl ontologies. *Semantic Web*. 2011; 2(1): 11–21.