International Journal on Semantic Web and Information Systems
Volume 18 • Issue 1

Deep Embedding Learning With Auto-Encoder for Large-Scale Ontology Matching

Meriem Ali Khoudja, LRDSI Laboratory, Faculty of Science, University of Blida 1, Algeria*
Messaouda Fareh, LRDSI Laboratory, Faculty of Science, University of Blida 1, Algeria
Hafida Bouarfa, LRDSI Laboratory, Faculty of Science, University of Blida 1, Algeria

ABSTRACT

Ontology matching is an efficient method to establish interoperability among heterogeneous ontologies. Large-scale ontology matching still remains a big challenge for its time and large memory space consumption. The actual solution to this problem is ontology partitioning, which is also challenging. This paper presents DeepOM, an ontology matching system, to deal with this large-scale heterogeneity problem without partitioning using deep learning techniques. It consists of creating semantic embeddings for concepts of input ontologies using a reference ontology and uses them to train an auto-encoder in order to learn more accurate and less dimensional representations for concepts. The experimental results of its evaluation on large ontologies and its comparison with different ontology matching systems which have participated to the same test challenge are very encouraging with a precision score of 0.99. They demonstrate the higher efficiency of the proposed system to increase the performance of the large-scale ontology matching task.

KEYWORDS

Auto Encoder, Large-Scale Ontology Matching, Ontology Alignment, Semantic Concepts' Embeddings, Semantic Web, Unsupervised Deep Learning

INTRODUCTION

Ontologies are the cornerstone of the semantic web. They provide representing, sharing and reuse of knowledge as a communication tool for applications developed in different ways. An ontology is an explicit description of the concepts, properties, relationships and individuals that may exist for a particular domain. It reflects knowledge from a certain domain of discourse (Zamazal, 2020). However, most applications require access to multiple ontologies. In addition, the construction of ontologies by various experts leads to heterogeneity at different levels, due to the rapid development of the semantic web.

Therefore, it is very interesting to identify correspondences between semantically related entities of heterogeneous ontologies. That allows agents using different ontologies to inter-operate. These correspondences, called alignment or mapping, are the backbone of the ontology matching task, which is the promising solution to this ontology semantic heterogeneity problem. In addition, automatic and

DOI: 10.4018/IJSWIS.297042 *Corresponding Author

semi-automatic ontology matching techniques should be developed in order to reduce the burden of its manual creation and maintenance (Khiat & Benaissa, 2015).

Likewise, ontologies of most applications (like in medicine and astronomy) are of big size. And, large ontologies include a high conceptual heterogeneity. That could decrease the efficiency of ontology matching systems facing other challenges as shortage of memory and long-time processing. Such issue makes scaling up the ontology matching process a very interesting problem.

Deep learning techniques have been recently used to address important problems in many research axes, such as image processing, natural language processing, information retrieval, signal processing and many others as in (Gupta et al., 2019; Sedik et al., 2021; Al-Smadi et al., 2018). Sophisticated artificial intelligence systems use deep learning to solve computational tasks and complex problems quickly (Fiorini, 2020). These techniques are very appropriate for dealing with large datasets. They have the ability to analyse and interpret massive amounts of data, that require efficient and effective computational tools.

Although deep learning techniques are very appropriate for dealing with large datasets, they have limited use in ontology matching. Even the few approaches that employ these computational models aim at enhancing the performance of the ontology matching task, and not at handling the large-scale heterogeneity problem (Portisch & Paulheim, 2018; Chang et al., 2019; Hertling & Paulheim, 2018; Monych et al., 2020; Roussille et al., 2018). Then, they tested their methods on ontologies of small sizes. The commonly promising solution for dealing with the large-scale ontology matching issue is partitioning (Tran et al., 2012; Laadhar et al., 2019; Laadhar et al., 2018; Jiménez-Ruiz et al., 2018; Balachandran et al., 2019). It consists on dividing the input ontologies into several sub-ontologies. The overall result of matching is obtained after combining the individual results of matching sub-ontologies. However, the partitioning phase is also challenging; The method of dividing input ontologies, the number of resulted partitions, sizes of partitions, and all parameters of the partitioning process are delicate to define as well. Moreover, several semantic links inside ontologies are expected to be lost when dividing input ontologies. That affects the matching quality.

In this paper, the authors address these challenges and propose a new ontology matching system, called DeepOM, for automatically matching large ontologies without partitioning and basing on deep learning techniques. DeepOM firstly extracts the requisite ontological information from input ontologies and pre-process it. A reference ontology is then used to transform ontological concepts into numerical vectors that deep learning models can use as input. Auto encoders are common deep learning models. They are great at representation learning. Once the semantic embeddings for concepts are created, they can be used to train an auto-encoder, in order to output finer and smaller representations for ontological concepts. After that, the cosine similarity is used to compute similarities between the compact vectorial representations of concepts. Finally, a filtering process is applied using a defined alignment threshold, in order to keep only the most appropriate correspondences that compose the final mapping. The main contributions of this work are:

- Employing deep learning techniques in order to effectively match large-scale ontologies without partitioning them, and at the lowest time process and memory space cost.
- Representing the concepts of input ontologies in a multi-dimensional embedding space, using a smaller and well selected reference ontology. That aims for perfecting the matching performance and reducing its complexity.
- Training an auto-encoder on the concepts' embeddings, in order to learn more accurate and more compact representations for input concepts. That leads to better performance and less complexity as well.
- Unsupervised learning doesn't require a learning base, which necessitates a delicate process to prepare.

- The results of evaluating DeepOM on large OAEI ontologies, and its comparison with ontology matching systems which have participated to the same test case, demonstrate its high ability to tackle the large-scale problem.

The remainder of this paper is organized as follows. After this introduction, the authors first review and study the existing work in the area of ontology matching, particularly of large-scale ontology matching and matching based on deep learning techniques. And, they discuss their advantages and limits, to figure out contributions of this work. Next, they present the detailed workflow of the proposed ontology matching system that they propose. Then, they describe the evaluation of DeepOM, conducted on the Anatomy track from the 2020 campaign of the open Ontology Alignment Evaluation Initiative (OAEI). The authors present and analysis the results of these experiments and discuss the performance of their system. Finally, they conclude this paper and outline their future work.

RELATED WORK

Ontology matching is a core issue for enabling interoperability between heterogeneous ontologies. For that object, several matching approaches have been proposed in the scientific literature. Ontology matching process is generally based on measuring similarity between entities of the concerned ontologies. On this basis, that numerous classifications of the various ontology matching approaches are given as in (Shvaiko & Euzenat, 2011; Otero-Cedeira et al., 2015; Thiéblin et al., 2020).

Large-Scale Ontology Matching

Nowadays, ontologies of numerous domains are very large. Biomedical ontologies have attacked much attention. Thus, matching such ontologies becomes a very complex task, especially in terms of time processing and allocated memory space. Ontology matching approaches which address this problem usually partition the large ontologies into small sub-ontologies in order to reduce the matching space. Babalou et al. (2016) proposed to classify the different partitioning methods into the following categories: modulation, clustering, summarization, decomposition, and divide and conquer. In the following, some work of different partitioning-based ontology matching techniques is presented:

- Tran et al. (2012) proposed a partitioning approach to break up the large matching problem into smaller matching sub-problems. They first semantically split anatomy ontology into groups called clusters. Basing on the information content of their concepts, and a scalable agglomerative hierarchical clustering algorithm. They use then a filtering method to select the possible similar partitions in order to reduce the computation time.
- The study of Laadhar et al. (2019) presented a local matching learning strategy to align large and complex biomedical ontologies, combining ontology partitioning with machine learning techniques. It defines a new partitioning approach, based on the hierarchical agglomerative clustering. This latter breakup large ontology alignment task into a set of local sub-matching tasks. Instead of defining a global machine learning model for the entire ontology matching task, it performs a machine learning model for each local sub-matching task and provides its corresponding training set, which is automatically generated by exploring the external biomedical knowledge bases without any gold standard or user involvement. Therefore, each proposed local matching learning model automatically provides adequate matching parameters for every local sub-matching task.
- The approach proposed by Jiménez-Ruiz et al. (2018) consists on splitting the ontology matching task into smaller and more tractable matching subtasks, basing on a lexical index and locality modules. Two clustering strategies are presented for the lexical index. Naive strategy relies on a simple splitting method, to randomly divide entries into a given number of clusters of the same

size. And, Neural embedding strategy relies on a log-linear neural embedding model. It aims to reduce the global size of the computed division of the matching task.

- The study of Balachandran et al. (2019) is based on graph partitioning to improve the execution time of ontology mapping process. The proposed ontology mapping process works in three consecutive phases. First, the cluster-walktrap methodology is used to partition the ontologies into sub-ontologies and identify the correspondence between the concepts in parallel. Then, the factored ontologies are represented in vector space model and similarities are computed between concepts. Finally, a collaborative decision on the mappings is generated, taking into account the similarities of the previous phase.
- Laadhar et al. (2018) proposed an approach that applies the hierarchical agglomerative clustering technique to divide an ontology into a set of partitions. Then, it uses an automated tuning process, which generates the adequate thresholds of the available similarity measure for any biomedical matching task.

Deep Learning for Ontology Matching

In the last decades, artificial neural networks have been widely used to tackle the ontology matching problem (Ali Khoudja et al., 2018b; Djeddi & Khadir, 2013; Ali Khoudja et al., 2018a). However, the use of deep neural networks has not attracted much attention from research teams to match heterogeneous ontologies, despite the fact that ontology matching is an active field of current research. Some of these approaches are presented in the following:

- Xiang et al. (2015) proposed ERSOM, an ontology matching approach which mainly uses the deep neural network model to learn the high-level abstract representations of classes and properties from their descriptions, and an iterative similarity propagation method based on more abundant structure information of the ontology, for ontology matching in an unsupervised way. Qiu et al. (2017) added a supervised learning step when training data is available to refine the learned representation, and then allowed to learn the representation of ontology entity in the cases the training data exists or not.
- The work of Liu et al. (2019) presented a new ontology mapping system called HISDOM, which uses comprehensive factors like concept names, attributes, instances, and structural similarities to determine the similarity of ontology. And then dynamically derives the weight of those different factors in the overall ontology similarity proportional to the amount of information of each factor in the ontology, to determine whether the two ontologies have mapping relationships. HISDOM also uses a convolutional neural network to extract and calculate the comment and annotation semantics and find their similarity according to the extent of annotation.
- The study of Nkisi-Orji et al. (2018) introduced a random forest classifier for ontology alignment which integrates semantic similarity features, string-based similarity features and semantic context features, using word embedding. It completes alignment in two stages. It first selects a set of candidate alignments using basic matching techniques. After that, a machine classifier determines the true alignments from entity pairs of the candidate alignments, using feature vectors that are generated from a variety of direct and indirect similarity indicators.
- Dhouib et al. (2019) proposed a new ontology alignment approach inspired by an existing proposal (Zhang et al., 2014). It combines the radius measure and word embedding. They consider word embedding to get a vector representation of the concepts to be matched, and use it to compute hierarchical relations between concepts.
- The approach proposed by Chandrashekar et al. (2018) aims to discover the relationships between concepts from the analysis of semantic features across multiple ontologies, and identify the abstractions of the ontological relationships through mapping between features to the ontologies.

The ontology mapping is performed through ontology search, feature extraction and word embeddings.

Analysis of Related Work

The study of the related work allows us to identify their limits, and outline contributions of this study. Large-scale ontology matching still presents a real challenge because it is a time consuming and memory intensive process. Deep learning techniques are not commonly used to tackle this problem, although they are very appropriate for dealing with large datasets. Even the small amount that employ these models aim at enhancing the performance of the ontology matching task, and not at handling the heterogeneity between large ontologies. Besides, they tested their methods on ontologies of small sizes. Partitioning large-scale ontologies is the commonly promising solution to deal with this issue. Ontology matching techniques that deal with this problem search to divide the large-scale ontology matching task into small-scale and more tractable matching sub-tasks, as presented above. However, partitioning ontologies also presents a problem. It may decrease in most cases the matching quality, owing to the fact that, several semantic links inside ontologies are expected to be lost. Furthermore, the number of partitions, size and elements of each partition, how to align these divisions, are also challenging and affect the matching performance. The authors aim, by ***DeepOM***, the system that they propose in this paper, to employ deep learning techniques in order to efficiently match huge ontologies without partitioning them, and at the lowest time process and memory space cost.

DEEP ONTOLOGY MATCHING

The idea behind DeepOM is to automatically treat the large-scale ontology matching issue in two stages. At each stage, it seeks for providing more representative and less dimensional real-valued vectors for concepts of input ontologies. First, it creates semantic embeddings for ontological concepts, basing on the semantic similarity between them and the concepts of a smaller and well selected reference ontology. That perfects the matching process, because of the fact that, each concept is represented in a vector space of a wide number of dimensions, where each value adds more precision to the concerned concept. And, it reduces the matching complexity, since, manipulating vectors of real values instead of ontological concepts is much less complicated. Second, DeepOM trains an auto-encoder on the generated concepts' vectors, in order to learn high-level and more compact embeddings for input concepts. This learning process also leads to better matching performance since, the auto-encoder keeps the most representative values for each concept. Also at this stage, compressing concepts' embeddings to a lower-dimensional vector space decreases the large-scale matching complexity.

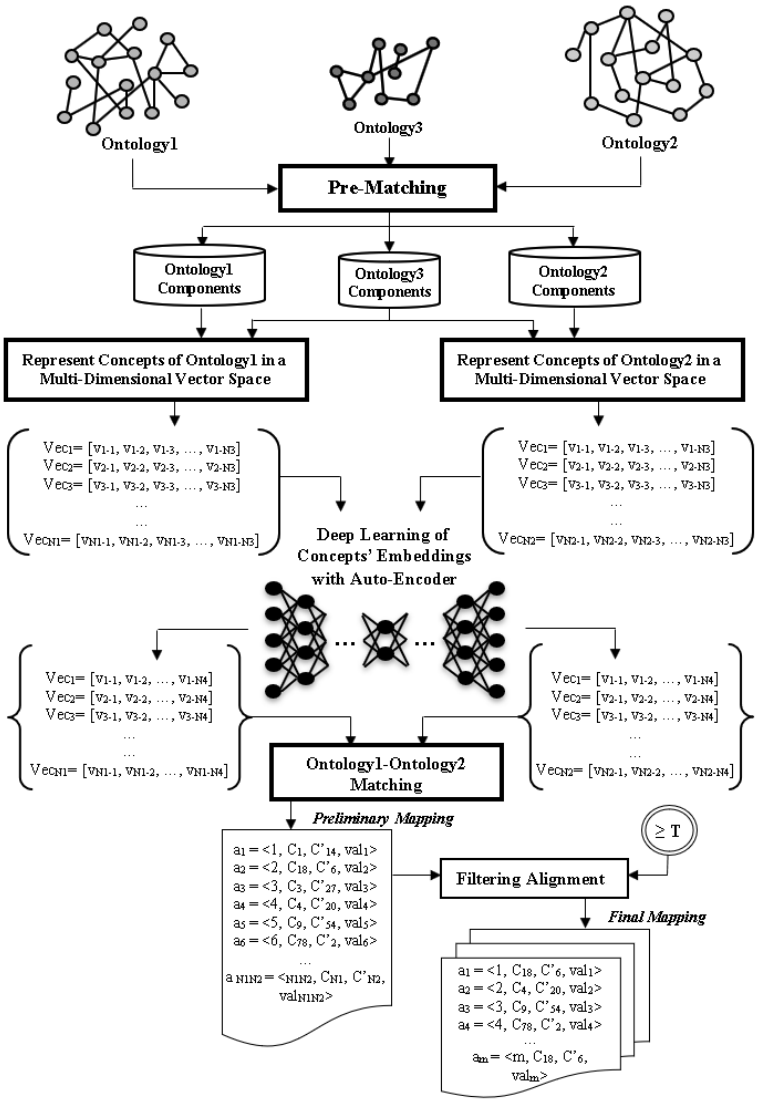
The processing workflow of DeepOM is illustrated in Figure 1. It could be summarized in four major phases: *Pre-Matching Phase, Embedding Phase, Deep Learning Phase* and *Matching Phase.*

Matching two given ontologies *Ontology1* and *Ontology2* is the process of finding a set of *m* correspondences (alignment) $A = \{a_1, a_2, a_3, \ldots, a_m\}$. Each correspondence $a_{i,\,(i=1\text{-}m)}$ is defined by a quadruple as: $a_i = < id_i, C, C', val_i >$. Where: id_i is the correspondence identifier; C is a concept from *Ontology1*; C' is a concept from *Ontology2*; and val_i the correspondence value between C and C' provided by DeepOM. This latter is in range [0,1], and reflects the similarity measure between the linked concepts.

Step 1: Pre-Matching

The first step aims to prepare ontologies for matching. It is such an important process, since the input ontologies are heterogeneous and different in their components' availability and entities' lexicon for the authors' interests. The authors pre-match input ontologies in two main sub-steps:

Figure 1. Processing Workflow of DeepOM



1. Extracting Ontological Information

This task consists on loading the ontologies needed for matching, and extracting their components which are necessary for generating alignment. An ontological concept is defined by three main aspects: *Term*, *Intention* and *Extension*. As the authors aim to perform the matching process in a complete way, they cover the three main dimensions of concepts. They extract for each concept *C* from *Ontology1* and *Ontology2* its:

- ***Lexical label***, which is the representative term used to describe that concept;
- ***Related concepts***, which are the concepts from the same ontology that are related to the concerned concept. Concepts of ontologies are related to each other with distinct types of relationships.

The most basic type of relations in an ontology is the *subsumption* relation, also known as *is-a* relation. It provides the tree-like taxonomy of the ontology. By this relation, an ontological concept has principally a ***parent-concept***, a ***child-concept*** and a ***sibling-concept***. According to this structure, three main types of related concepts are extracted:

- ***Ancestors***, which are the elements of the set of concepts composed by, the parents of *C*, and the parents of their parents, along the path to root. i.e., the authors extract all parent-concept levels of the concept in question;
- ***Descendants***, which are the direct child-concepts of *C*. As a concept has a significant number of child-concepts compared with parent-concepts, the authors find that the first child-concept level is sufficient to have related descendants' concepts;
- ***Siblings***, which are the direct child-concepts of the direct parent-concepts of *C*. i.e., concepts of the first child-concept level of the first parent-concept level of the concerned concept are extracted;

- ***Individuals***, which represent the instance-level of the concept, described by its concrete objects.

2. Pre-Processing of Ontological Components

In this sub-step, the authors mainly interest in pre-processing the lexical information extracted from input ontologies. Thus, once the ontological components are extracted, they analyze and process, for each concept, its label, individuals' names, as well as the labels of its related concepts (ancestors, descendants and siblings). Considering an extracted textual information T to be pre-treated. The pre-processing task outputs a set of processed terms. It is performed as present the following points:

- **Tokenization:** Consists on segmenting T to a set of tokens according to space (' ') and two types of dashes ('-' and '_').
- **Removing stop words:** Consists on removing the commonly used words which do not carry useful information for matching. For that, the authors use the English **nltk**[1] stop-words list.
- **Denoising:** Aims to get rid of unhelpful elements of the textual information. In the case of this study, it consists on lowercasing all characters, removing tokens of length 1 (excepting numbers) and removing punctuations marks as well as all special and non-ASCII characters.

Step 2: Create Semantic Embeddings for Concepts

This step consists on transforming the concepts of input ontologies into vectorial representations that deep learning models can use as input. the authors use another ontology, called ***reference*** ontology, in order to represent each concept from *Ontology1* and *Ontology2* in a numerical multi-dimensional vector space.

For the following, the set of *Ontology1*'s concepts is defined by $C = \{c_{i,\ i = 1\text{-}N1}\}$, the set of *Ontology2*'s concepts by $C' = \{c'_{i,\ i = 1\text{-}N2}\}$, and the set of concepts of the reference ontology by $C'' = \{c''_{i,\ i = 1\text{-}N3}\}$, where N1, N2 and N3 denote the number of concepts of *Ontology1*, *Ontology2* and the reference ontology respectively.

Algorithm 1 demonstrates the task of this step. It is performed based on computing similarities between concepts of the reference ontology and elements of C and C'. the authors represent each concept from *Ontology1* and *Ontology2* by a vector of N3 numerical values. Each value is the similarity score between the concerned concept and a concept from the reference ontology. Since the size of the reference ontology is N3, all vectorial representations of concepts of *Ontology1* and *Ontology2* are of length N3 for each vector. i.e., concepts of input ontologies are represented in a N3-dimensional vector space. As results of this step, the embedding representations of ontologies' concepts are created after that the execution of the algorithm is completed.

Algorithm 1. Create concepts' embeddings for input ontologies

```
Input:      Ontology1's concepts: C = {c_i, i = 1-N1};
            Ontology2's concepts: C' = {c'_i, i = 1-N2};
            Reference ontology's concepts: C" = {c"_i, i = 1-N3}.
Begin
// Vectorial representations for elements of C
Initialization of V = {}
for i from 1 to N1
      Initialization of vec_i = []
      for j from 1 to N3
            vec_i[j] = Semantic similarity value between c_i and c"_j
      append vec_i to V
// Vectorial representations for elements of C'
Initialization of V' = {}
for i from 1 to N2
      Initialization of vec'_i = []
      for j from 1 to N3
            vec'_i[j] = Semantic similarity value between c'_i and c"_j
      append vec'_i to V'
End
Output:     Embeddings of C: V = (vec_i, i = 1-N1), vec_i = [v_j, j = 1-N3];
            Embeddings of C': V' = (vec'_i, i = 1-N2), vec'_I = [v'_j, j = 1-N3].
```

The accuracy of the generated embeddings is highly dependent upon two major factors.

1. Reference Ontology Definition

The reference ontology has a great impact on the performance of the matching process. Thus, its determination is such a delicate and careful task. It depends on input ontologies and should be:

- Of the same domain as *Ontology1* and *Ontology2*, and semantically close to them. Otherwise, the embedding vectors would be overpowered by zeros. Thus, they would not provide the real representations for concepts.
- In the-middle-of-the-road between *Ontology1* and *Ontology2*. i.e., it should be neutral and balanced between them, so as to afford fair concepts' representations.
- Of an appropriate size. i.e., it must not be very small, not useful, nor larger than input ontologies, so matching gets more complicated.

2. Similarity Measurement

The adequacy of the numerical values of the concepts' embeddings relies on how similarities are computed between input ontologies and the reference ontology. As this is a very exact task, the authors seek for performing it on a high level of accuracy. They exploit the different aspects of ontological concepts. Therefore, they proceed and combine several matchers:

- **Terminological matcher:** Exploits semantics inside concepts' lexicon. It measures both *context-based* similarity and *syntactical* similarity. And, it combines them in a way that, the weight

assigned to each element reflects its accurate need proportion in that current case. The authors propose the following formula to compute the terminological similarity between two concepts C_1 and C_2:

$$TerSim(C_1, C_2) = \frac{2 * \left|Label_1 \cap Label\right| + D_1 + D_2}{\left|Label_1\right| + \left|Label_2\right|} \quad (1)$$

where $Label_1$ is the pre-processed label of C_1; $Label_2$ is the pre-processed label of C_2; D_1 is the similarity report of $\left(Label_1 - Label_2\right)$ compared to $\left(Label_2 - Label_1\right)$; D_2 is the similarity report of $\left(Label_2 - Label_1\right)$ compared to $\left(Label_1 - Label_2\right)$. For each pair of individual terms, the authors take the maximum similarity between the two values provided by an ***external knowledge resource*** and ***Jaro measure***.

- **Structural matcher:** Measures the similarity between concepts basing on their structure, which refers to their related concepts. The authors use the following formula to compute structural similarities between ancestors, descendants and siblings of C1 and C2:

$$StrSim(C_1, C_2) = \frac{2 * \left|RC_1 \cap RC_2\right| + D_1 + D_2}{\left|RC_1\right| + \left|RC_2\right|} \quad (2)$$

where RC_1 is the set of concepts related to C_1; RC_2 is the set of concepts related to C_2; D_1 is the similarity report of $\left(\boldsymbol{RC}_1 - \boldsymbol{RC}_2\right)$ compared to $\left(\boldsymbol{RC}_2 - \boldsymbol{RC}_1\right)$; D_2 is the similarity report of $\left(\boldsymbol{RC}_2 - \boldsymbol{RC}_1\right)$ compared to $\left(\boldsymbol{RC}_1 - \boldsymbol{RC}_2\right)$. This similarity report is computed using the terminological similarity equation (Equation.1) for each pair of individual related concepts.

Related concepts (RC_1 for C_1 and RC_2 for C_2) refers to sets of ancestors, descendants and siblings of C_1 and C_2 for each case.

- **Extensional matcher:** The authors measure the similarity between instances of C_1 and C_2 using the Jaccard similarity, given by formula:

$$ExtSim(C_1, C_2) = \frac{\left|Inst_1 \cap Inst_2\right|}{\left|Inst_1 \cup Inst_2\right|} \quad (3)$$

where $Inst_1$ is the set of instances of C_1 and $Inst_2$ is the instances set of C_2.

Combining the individual similarity measures is necessary in order to get the final semantic similarity between C_1 (from C+C') and C_2 form (C''). Unlike other ontology matching techniques, which give equal weights for similarity values, DeepOM combines them in an additive way that, the lack individuals and related elements for some concepts would not affect the matching performance. Algorithm 2 demonstrates this process.

Algorithm 2. Measure semantic similarity between two given concepts

```
Input:      C1: Concept from Ontology1+Ontology2: Set of N1
            concepts: C + Set of N2 concepts: C', C = {ci, i = 1-N1},
            C' = {c'i, i = 1-N2};
            C2: Concept from Reference ontology: Set of N3
            concepts: C", C" = {c"i, i = 1-N3};
            Sim_Threshold.
Begin
// Computing similarities between C1 and C2
      Ins1 = list of c1 Instances
      Anc1 = list of c1 Ancestors
      Des1 = list of c1 Descendants
      Bro1 = list of c1 Siblings
            Ins2 = list of c"2 Instances
            Anc2 = list of c"2 Ancestors
            Des2 = list of c"2 Descendants
            Bro2 = list of c"2 Siblings
            TerVal = TerSim(c1,c"2)
            AncVal = StrSim(c1,c"2) \\ RC1 == Anc1; RC2 == Anc2
            DesVal = StrSim(c1,c"2)  \\ RC1 == Des1; RC2 == Des2
            BroVal = StrSim(c1,c"2)  \\ RC1 == Bro1; RC2 == Bro2
            ExtVal = ExtSim(c1,c"2)  \\ Inst1 == Ins1; Inst2 == Ins2
// Combining similarities between C1 and C2
StrVal = Average(AncVal,DesVal,BroVal)
If (StrVal>Sim_Threshold) and (ExtVal>Sim_Threshold):
      SemVal = Average(TerVal,StrVal,ExtVal)
Else: If (StrVal>Sim_Threshold):
            SemVal = Average(TerVal,StrVal)
       Else: If (ExtVal>Sim_Threshold):
                  SemVal = Average(TerVal,ExtVal)
             Else:
                  SemVal = TerVal
End
Output:     SemVal: Semantic Similarity value between two given concepts;
```

Step 3: Deep Ontology Matching With Auto-Encoder

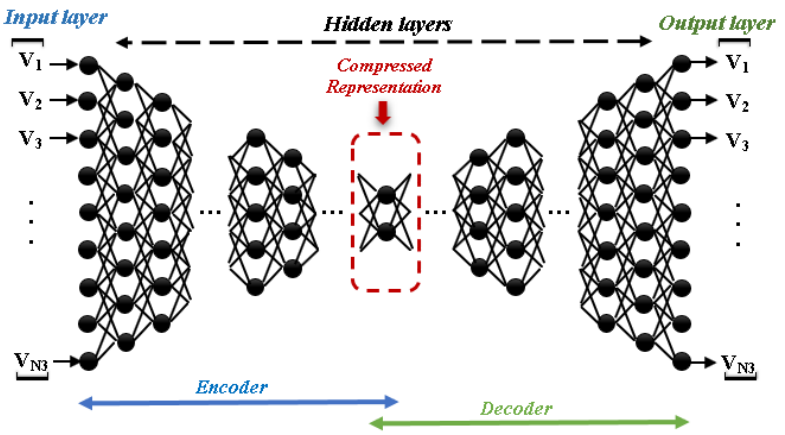
This step consists on using deep learning techniques to learn high-level embeddings for concepts of the two ontologies. This task aims to provide ***more accurate*** and ***less dimensional*** representations for concepts. That perfectly represents the input ontologies in an unsupervised way.

Auto-encoders seem to be very appropriate for such purposes. They are capable of creating sparse representations of the input data. Therefore, they can be used to compress the concepts' vectors resulted from the previous step, and represent them in a latent space.

Figure 2 presents the architecture of the auto-encoder model of DeepOM. It is a deep neural network with multiple layers. The output layer has the same dimension as the input layer. And, the architecture between them is mirrored. The model has two components: ***Encoder*** and ***Decoder***. The encoder compresses the input data into a lower dimension. Then, the decoder uses the compact representations to recreate the original input.

The number of layers of the deep network, the number of nodes per each layer, the number of training epochs, as well as the other auto-encoder parameters are fixed by trial-and-error. For training

Figure 2. Architecture of the Auto-Encoder Model



the auto-encoder model, the authors use the back-propagation learning method. They take the current concepts' vectorial representations as input. Then, they train the model to learn weights so as to compress down these vectors into a lower dimensional space. The final learned representations (of size N4 as shown in Figure 1) keep the most important features of ontological concepts.

Step 4: Generate Ontology1-Ontology2 Alignment

This final step is performed in two sub-steps.

1. Measuring Embeddings Similarity

Generating alignment between *Ontology1* and *Ontology2* requires computing similarities between their concepts. Since those concepts are represented by numbers in a vector space, the matching process consists on computing similarities between the corresponding vectors. For that, the authors use the cosine similarity measure defined by:

$$\mathrm{Cos}(\vec{x},\vec{y}) = \frac{\vec{x}.\vec{y}}{\|\vec{x}\|.\|\vec{y}\|} \tag{4}$$

2. Pruning Generated Alignment

Once similarities between concepts of the input ontologies have been measured, they undergo a filtering procedure, so as to keep only significant correspondences. For example, the authors do not consider a concepts' pair of which the value is equal to 0.0 as a correct correspondence. For that, they define an alignment threshold T, fixed by trial-and-error, to extract the final mapping, for which the similarity values exceed T. The authors aim by this task at improving the matching accuracy by removing irrelevant correspondences of low similarity scores, and keeping only the most appropriate correspondences.

EXPERIMENTS

In order to evaluate their ontology matching system, the authors proceed an experimental procedure described as follows.

Experimental Design

OAEI[2] is an international initiative which aims to evaluate ontology matching systems using diverse types of test ontologies. It is the most authenticated initiative in this scope. It offers various test sets aiming to compare the different participant systems on the same basis.

In this paper, the authors evaluate their system according to OAEI's Anatomy track. Zhang and Bodenreider (2007) discussed the challenges and issues and recapitulate their experience in aligning large-scale anatomical ontologies. At present, OAEI'2020 is the most recent OAEI campaign. As the authors look for passing the ontology matching task to the large scale, Anatomy track proposes test ontologies of appropriate sizes. The OAEI'2020 Anatomy track[3] comprises a single real-world test case about matching two fragments of biomedical ontologies describing the human anatomy and the anatomy of the mouse. The task is situated in a domain where we find large and carefully designed ontologies which are described in technical terms. The evaluation is based on a manually curated reference alignment.

The input ontologies to be matched are: **human** ontology with 3304 classes and **mouse** ontology with 2744 classes. Their concepts have no individuals (instances). They are represented in OWL language.

As reference ontology, the authors use the Anatomical Entity Ontology: **AEO**[4], an OWL ontology of anatomical structures with 250 classes. It expands the Common Anatomy Reference Ontology (CARO)[5], which is an upper-level ontology to facilitate interoperability between existing anatomy ontologies for different species. AEO is intended for being useful in increasing the knowledge amount of anatomy ontologies, facilitating annotation and in enabling interoperability across anatomy ontologies.

The structure of the trained auto-encoder model is [250-200-150-100-150-200-250]. The size of the reference ontology is 250. That generates a vectorial representation of 250 numbers for each concept. It is token as input and output to the network.

For the external resource, which is required for measuring the terminological similarity, the authors use **BioWordVec** (Zhang et al., 2019). It is an open set of biomedical word embeddings of 2,324,849 distinct words. It combines sub-word information from unlabeled biomedical text with MeSH[6], a widely-used biomedical controlled vocabulary.

To evaluate their ontology matching system, the authors use the standard evaluation measures: *precision*, *recall* and *F-measure* defined in the following. Precision and recall are the most widely used criteria for such evaluation. They are based on the comparison of the resulted alignment *A* against a reference alignment *R*. Precision and recall are inversely proportional. For that, their harmonic mean, F-measure, is also commonly used.

Definition 1: *Precision* measures the ratio of relevantly selected correspondences over the total number of selected correspondences as in formula (5):

$$Precision = \frac{\left|A \cap R\right|}{\left|A\right|} \tag{5}$$

Definition 2: *Recall* measures the ratio of relevantly selected correspondences over the total number of relevant correspondences as in formula (6):

$$Recall = \frac{\left|A \cap R\right|}{\left|R\right|} \tag{6}$$

Definition 3: *F-measure* aggregates precision and recall evenly as in formula (7):

$$F - measure = 2 * \frac{precision * recall}{precision + recall} \tag{7}$$

Experimental Results

In order to study the efficiency of DeepOM, the authors compare its results with the results[7,8,9] of the new systems participant to the same matching challenge for the three most recent OAEI campaigns. The results of the performed evaluation are summarized in Table.1. The best scores for each evaluation measure are marked in bold. The OAEI systems with which DeepOM is compared are: ATBox (Hertling & Paulheim, 2020), OntoConnect (Chakraborty et al., 2020), ALOD2Vec (Portisch & Paulheim, 2020), FCAMap-KG (Chang et al., 2019), DOME (Hertling & Paulheim, 2018), DESKMatcher (Monych et al., 2020), Holontology (Roussille et al., 2018) and AGM (Lütke, 2019). StringEquiv is a baseline of OAEI that generates alignment basing on exact string matching of concepts' labels. They are classified in Table 1 decreasingly according to their F-measure results.

As the aim, by DeepOM, is to both maximize the matching quality and minimize the large-scale matching complexity, the authors evaluate the proposed ontology matching system at two stages.

1. Evaluate the Matching Quality

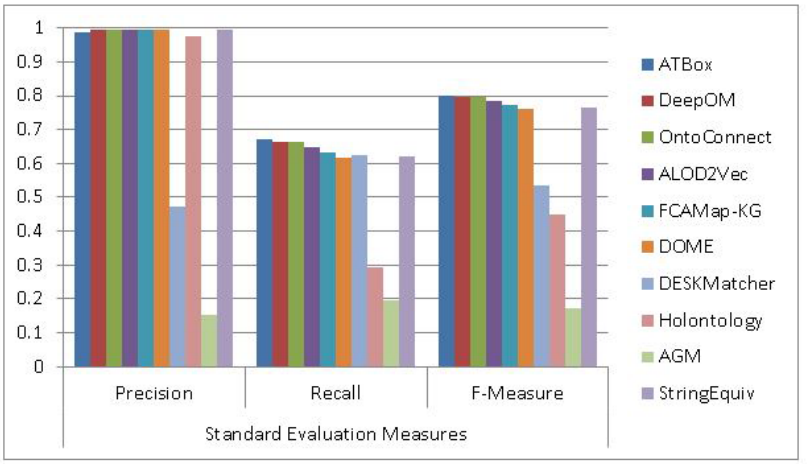
Graphs in Figure 3 outline the evaluation results in terms of the three standard evaluation metrics defined in the precedent sub-section.

We can see from Figure 3 that, 1. DeepOM presents an excellent precision score (0.994). It is among the five top ranked systems of which the values exceed 0.99, and outperforms the other matching systems in terms of precision. 2. Regarding recall, DeepOM achieved a good value (0.665). It presents (with OntoConnect) the second-best score after ATBox with a slight difference of 0.006, and outperforms the other systems. 3. For F-measure, which combines precision and recall evenly, the system proposed in this work presents a high score (0.797), greater than the baseline (StringEquiv) which is based on normalized string equivalence. DeepOM is among the top three ranked systems that exceed 0.79. The other matching systems present competitive results excepting DESKMatcher, Holontology and AGM.

Table 1 Evaluation results for Anatomy'20 track

System	Standard Evaluation Measures			Runtime(s)
	Precision	Recall	F-Measure	
ATBox	0.987	0.671	0.799	192
DeepOM	0.994	0.665	0.797	149
OntoConnect	0.996	0.665	0.797	248
ALOD2Vec	0.996	0.648	0.785	75
FCAMap-KG	0.996	0.631	0.772	25
DOME	0.997	0.615	0.761	22
DESKMatcher	0.472	0.623	0.537	391
Holontology	0.976	0.294	0.451	265
AGM	0.152	0.195	0.171	628
StringEquiv	0.997	0.622	0.766	6

Figure 3 Evaluation results for OAEI-Anatomy track



2. Evaluate the Matching Complexity

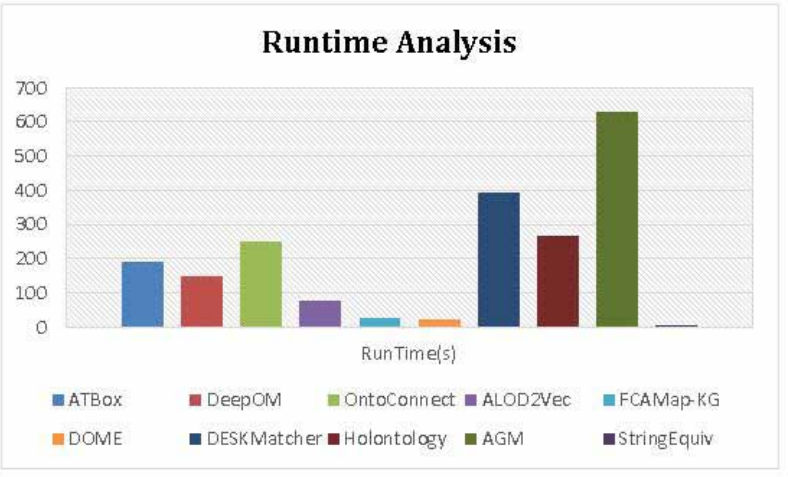
As the authors really interest in reducing the matching complexity, they compare the matching runtime required by their system with the other participant matching systems. Figure 4 illustrates this comparison.

We can observe from Figure 4 that, DeepOM performs the ontology matching process at short notice comparing with the other matching systems. It is among the 4 systems out of 9 that are able to achieve the matching task in less than 150 seconds. These are DeepOM, ALOD2Vec, FCAMap-KG and DOME which has the shortest runtime.

Experimental Summary

According to the previous results, we can conclude the following. As the authors aim by DeepOM to both increase the matching quality and decrease the matching complexity, they analyse results of F-measure, which reflects the real quality of matching, with respect to the matching runtime:

Figure 4. Runtime analysis for OAEI-Anatomy track



- The preliminary results of DeepOM are very promising. It has achieved a score of 0.8 for F-measure, which is a very good value for the ontology matching task at least as a start.
- Comparing DeepOM with different matching systems which have participated to the OAEI-Anatomy track, it outperforms them in terms of F-measure, excluding ATBox and OntoConnect.
- As for these two systems, DeepOM has matching results similar to OntoConnect. The results of ATBox are a bit higher but with a very slight difference (0.002). Regarding complexity, DeepOM has the shortest runtime.

The conducted evaluation of DeepOM demonstrates that, concepts' embeddings using the reference ontology and learning them with auto-encoder have improved the performance of the ontology matching task. Moreover, representing ontological concepts by numerical values in a vector space has efficiently solved the large-scale ontology matching problem.

The experimental results of DeepOM show a very good matching performance at lowest cost. This means that the authors have achieved the two objectives of this system, which are improving the matching performance and reducing its complexity. Therefore, they have effectively tackled the challenge of large-scale ontology matching.

CONCLUSION

The ontology matching field is maturing with enormous number of matching techniques. However, dealing with large ontologies still remains a key challenge. The works which deal with this problem are based on ontology partitioning which is also challenging. In this paper, the authors propose DeepOM, a large-scale ontology matching system based on deep learning techniques to deal with the large-scale heterogeneity problem without partitioning. The key novelty of DeepOM is to use a reference ontology to create semantic embeddings for ontological concepts, which are used to train an auto-encoder in order to learn more accurate and less dimensional representations for concepts.

The results of its evaluation on large OAEI ontologies, and its comparison with ontology matching systems participant to the same test case, show that, DeepOM outperforms the ontology matching baseline with high ability to tackle the large-scale problem. Learning concepts' embeddings using auto-encoder is effective for matching large-scale ontologies, and all the matching factors of DeepOM are positive towards improving the ontology matching quality.

Although the experimental results of this work are very encouraging, the authors aim, as future work, at filtering concepts of the reference ontology, and removing its isolated and useless concepts before generating the new representations for input ontologies. They also plan to test DeepOM on larger and more huge ontologies with many instances. In addition, they intend to detailly evaluate each step of DeepOM separately, and evaluate the generated alignment values using specific evaluation measures dedicated for such purposes.

REFERENCES

Al-Smadi, M., Qawasmeh, O., Al-Ayyoub, M., Jararweh, Y., & Gupta, B. (2018). Deep Recurrent neural network vs. support vector machine for aspect-based sentiment analysis of Arabic hotels' reviews. *Journal of Computational Science*, *27*, 386–393. doi:10.1016/j.jocs.2017.11.006

Ali Khoudja, M., Fareh, M., & Bouarfa, H. (2018a, October). A new supervised learning based ontology matching approach using neural networks. In *International Conference Europe Middle East & North Africa Information Systems and Technologies to Support Learning* (pp. 542-551). Springer.

Ali Khoudja, M., Fareh, M., & Bouarfa, H. (2018b, November). Ontology matching using neural networks: survey and analysis. In *2018 International Conference on Applied Smart Systems (ICASS)* (pp. 1-6). IEEE. doi:10.1109/ICASS.2018.8652049

Babalou, S., Kargar, M. J., & Davarpanah, S. H. (2016, April). Large-scale ontology matching: a review of the literature. In *2016 Second International Conference on Web Research (ICWR)* (pp. 158-165). IEEE. doi:10.1109/ICWR.2016.7498461

Balachandran, S., Ranganathan, V., & Vetriveeran, D. (2019). Aligning large biomedical ontologies for semantic interoperability using graph partitioning. *International Journal of Biomedical Engineering and Technology*, *31*(2), 137–160. doi:10.1504/IJBET.2019.102120

Chakraborty, J., Yaman, B., Virgili, L., Konar, K., & Bansal, S. K. (2020). OntoConnect: Results for OAEI 2020. *Ontology Matching*, 204.

Chandrashekar, M., Nagulapati, R., & Lee, Y. (2018, June). Ontology mapping framework with feature extraction and semantic embeddings. In *2018 IEEE International Conference on Healthcare Informatics Workshop (ICHI-W)* (pp. 34-42). IEEE. doi:10.1109/ICHI-W.2018.00012

Chang, F., Chen, G., & Zhang, S. (2019). FCAMap-KG Results for OAEI 2019. *OM@ ISWC, 2536*, 138-145.

Dhouib, M. T., Zucker, C. F., & Tettamanzi, A. G. (2019, September). An ontology alignment approach combining word embedding and the radius measure. In *International Conference on Semantic Systems* (pp. 191-197). Springer.

Djeddi, W. E., & Khadir, M. T. (2013). Introducing artificial neural network in ontologies alignement process. In *New Trends in Databases and Information Systems* (pp. 175–186). Springer. doi:10.1007/978-3-642-32518-2_17

Fiorini, R. A. (2020). Computational intelligence from autonomous system to super-smart society and beyond. *International Journal of Software Science and Computational Intelligence*, *12*(3), 1–13. doi:10.4018/IJSSCI.2020070101

Gupta, B. B., Agrawal, D. P., & Yamaguchi, S. (2019). *Deep learning models for human centered computing in fog and mobile edge networks*. Academic Press.

Hertling, S., & Paulheim, H. (2018). DOME results for OAEI 2018. *OM@ ISWC, 2288*, 144-151.

Hertling, S., & Paulheim, H. (2020, December). ATBox results for OAEI 2020. *CEUR Workshop Proceedings*, *2788*, 168–175.

Jiménez-Ruiz, E., Agibetov, A., Samwald, M., & Cross, V. (2018, December). We divide, you conquer: From large-scale ontology alignment to manageable subtasks with a lexical index and neural embeddings. *CEUR Workshop Proceedings*, *2288*, 13–24.

Khiat, A., & Benaissa, M. (2015). A new instance-based approach for ontology alignment. *International Journal on Semantic Web and Information Systems*, *11*(3), 25–43. doi:10.4018/IJSWIS.2015070102

Laadhar, A., Ghozzi, F., Ichise, R., Bousarsar, I. M., Ravat, F., & Teste, O. (2018, October). Partitioning and matching tuning of large biomedical ontologies. In *13th International Workshop on Ontology Matching co-located with the 17th International Semantic Web Conference (OM 2018)* (pp. 220-221).

Laadhar, A., Ghozzi, F., Megdiche, I., Ravat, F., Teste, O., & Gargouri, F. (2019, April). Partitioning and local matching learning of large biomedical ontologies. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing* (pp. 2285-2292). doi:10.1145/3297280.3297507

Liu, J., Tang, Y., & Xu, X. (2019, December). HISDOM: A Hybrid Ontology Mapping System based on Convolutional Neural Network and Dynamic Weight. In *Proceedings of the 6th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies* (pp. 67-70). doi:10.1145/3365109.3368779

Lütke, A. (2019). AnyGraphMatcher Submission to the OAEI Knowledge Graph Challenge 2019. *OM@ ISWC, 2536*, 86-93.

Monych, M., Portisch, J., Hladik, M., & Paulheim, H. (2020). DESKMatcher. *CEUR Workshop Proceedings*, *2788*, 181–186.

Nkisi-Orji, I., Wiratunga, N., Massie, S., Hui, K. Y., & Heaven, R. (2018, September). Ontology alignment based on word embedding and random forest classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 557-572). Springer.

Otero-Cerdeira, L., Rodríguez-Martínez, F. J., & Gómez-Rodríguez, A. (2015). Ontology matching: A literature review. *Expert Systems with Applications*, *42*(2), 949–971. doi:10.1016/j.eswa.2014.08.032

Portisch, J., & Paulheim, H. (2018). ALOD2Vec matcher. *OM@ ISWC, 2288*, 132-137.

Qiu, L., Yu, J., Pu, Q., & Xiang, C. (2017). Knowledge entity learning and representation for ontology matching based on deep neural networks. *Cluster Computing*, *20*(2), 969–977. doi:10.1007/s10586-017-0844-1

Roussille, P., Megdiche Bousarsar, I., Teste, O., & Trojahn dos Santos, C. (2018). Holontology: results of the 2018 OAEI evaluation campaign. *CEUR-WS: Workshop Proceedings*.

Sedik, A., Hammad, M., Abd El-Samie, F. E., Gupta, B. B., & Abd El-Latif, A. A. (2021). Efficient deep learning approach for augmented detection of Coronavirus disease. *Neural Computing & Applications*, 1–18. doi:10.1007/s00521-020-05410-8 PMID:33487885

Shvaiko, P., & Euzenat, J. (2011). Ontology matching: State of the art and future challenges. *IEEE Transactions on Knowledge and Data Engineering*, *25*(1), 158–176. doi:10.1109/TKDE.2011.253

Thiéblin, E., Haemmerlé, O., Hernandez, N., & Trojahn, C. (2020). Survey on complex ontology matching. *Semantic Web*, *11*(4), 689–727. doi:10.3233/SW-190366

Tran, D. T., Ngo, D. H., & Do, P. T. (2012, August). An information content based partitioning method for the anatomical ontology matching task. In *Proceedings of the Third Symposium on Information and Communication Technology* (pp. 272-281). doi:10.1145/2350716.2350757

Xiang, C., Jiang, T., Chang, B., & Sui, Z. (2015, September). Ersom: A structural ontology matching approach using automatically learned entity representation. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 2419-2429). doi:10.18653/v1/D15-1289

Zamazal, O. (2020). A Survey of Ontology Benchmarks for Semantic Web Ontology Tools. *International Journal on Semantic Web and Information Systems*, *16*(1), 47–68. doi:10.4018/IJSWIS.2020010103

Zhang, S., & Bodenreider, O. (2007). Experience in Aligning Anatomical Ontologies. *International Journal on Semantic Web and Information Systems*, *3*(2), 1–26. doi:10.4018/jswis.2007040101 PMID:18974854

Zhang, Y., Chen, Q., Yang, Z., Lin, H., & Lu, Z. (2019). BioWordVec, improving biomedical word embeddings with subword information and MeSH. *Scientific Data*, *6*(1), 1–9. doi:10.1038/s41597-019-0055-0 PMID:31076572

Zhang, Y., Wang, X., Lai, S., He, S., Liu, K., Zhao, J., & Lv, X. (2014). Ontology matching with word embeddings. In *Chinese computational linguistics and natural language processing based on naturally annotated big data* (pp. 34–45). Springer. doi:10.1007/978-3-319-12277-9_4

ENDNOTES

1 https://www.nltk.org/
2 http://oaei.ontologymatching.org/
3 http://oaei.ontologymatching.org/2020/anatomy/index.html
4 http://www.obofoundry.org/ontology/aeo.html

[5] http://www.obofoundry.org/ontology/caro.html
[6] https://www.ncbi.nlm.nih.gov/mesh/
[7] http://oaei.ontologymatching.org/2018/results/anatomy/index.html
[8] http://oaei.ontologymatching.org/2019/results/anatomy/index.html
[9] http://oaei.ontologymatching.org/2020/results/anatomy/index.html

Meriem Ali Khoudja is currently pursuing Ph.D. degree with the LRDSI Laboratory of Computer Science Department, Faculty of Sciences, Saad Dahleb University, Blida, Algeria. She got her baccalaureate in Mathematics in 2010. She received her Bachelor's degree in Computer Networks and Information Systems in 2013, and her Master's degree in Systems Engineering and Web Technologies in 2015 from Yahia Fares University, Medea, Algeria. Her research interest includes knowledge representation, ontology engineering, semantic web, ontology matching, artificial intelligence, machine learning and deep learning.

Messaouda Fareh is a doctor in computer sciences, and a lecturer at the computer sciences Department, University of Blida 1, Blida, Algeria. She is a member of the Laboratory LRDSI. Her research interests include ontology engineering and knowledge, information heterogeneity, uncertain knowledge of semantic web and data mining.

Hafida Bouarfa is a professor at University of Blida in Algeria. She obtained her engineer diploma in data processing in 1998 at the Data Processing Institute of Algiers. Her magister diploma has been prepared at the HEC of Montreal. In 2004, she obtained her PHD diploma. She is author of many research articles in the field of information systems, virtual organizations, knowledge management, collaborative work, etc.