Matching Ontologies with Word2Vec-Based Neural Network

Nikolay Teslya(✉) and Sergey Savosin

SPIIRAS, 14th line, 39, 199178 St. Petersburg, Russia
{teslya,SVSavosin}@iias.spb.su

Abstract. To date, knowledge engineering researchers have proposed a large number of ontology matching methods. In this paper, to solve the ontology mapping problem, it is proposed to use a vector language model based on the Word2Vec statistical model set. The vector model is implemented using a neural network based on the TensorFlow framework. The peculiarity of the method is the use of the baseline pre-trained Wor2Vec model based on texts from the English Wikipedia and Google News, which is consistently extended on the basis of relationships specific to the ontologies being matched. This approach allows the use of semantics of the language bypassing situations in which, due to the form of a word or a specific term, it is impossible to find a correspondence of ontology concepts.

Keywords: Ontology matching · Neural network · Word2Vec · Ontology

1 Introduction

To date, knowledge engineering researchers have proposed a large number of ontology matching methods. A large review of the methods was carried out in the book by Shvaiko and Euzenat "Ontology matching" [4]. In the classification of comparison methods proposed by them, two independent hierarchies are distinguished, the difference between which consists in considering the subject of comparison and the source of data based on the analysis of context (semantic and syntactic) and content (terminological, structural, extensional and semantic). The method proposed in this paper relates to syntactic methods for the investigation of ontology entities and according to another hierarchy, to the synthesis of semantic methods of context analysis with terminological methods of content analysis.

In the category of contextual matching methods, the main role is the analysis of the meanings of the terms used to label the ontology entities and the connections between them. The analysis is carried out using the top-level ontologies,

Funding: The reported study was funded by RFBR, project numbers 17-07-00327, 17-07-00328.

© Springer Nature Switzerland AG 2019
S. Misra et al. (Eds.): ICCSA 2019, LNCS 11619, pp. 745–756, 2019.
https://doi.org/10.1007/978-3-030-24289-3_55

linked data, dictionaries and thesauri. Methods of processing natural language are also used to bring the used forms of words to normal vocabulary forms, allowing to simplify the interaction with thesauri and dictionaries.

The extension of the considered methods is the use of background sources of knowledge, which include any information about the ontology and its individual concepts. There are several categories of sources of background knowledge - explicit (those that do not require additional processing of information used) and implicit (requiring additional processing to extract knowledge). Examples of explicit sources are Linked Data [7]; domain-specific ontologies [1]; common-purpose ontologies; classifiers; dictionaries and thesauri (WordNet [9]); lexical databases (WordNet, DANTE). Examples of implicit sources include ontology design patterns and ontology matching patterns [6,15,16,20].

Also there exists a scheme of the method based on the use of the problem area upper ontology as a source of background knowledge for ontology matching [18,20]. The method includes two steps: (1) anchoring the source and target concepts of the matched ontologies to the background knowledge ontology; (2) defining if the relationship between the source and target concepts exists by searching for a similar relationship in the background knowledge ontology. Paper [21] extends the results of previous works of the authors, considering the use of several ontologies as sources of background knowledge.

Background knowledge can also be used to train the neural network used in the ontology matching. Such networks most often refer to the type of classifying networks, as in the study of ontology concepts, it is necessary to cross-define the degree of proximity of the concepts of matched ontologies. For example, in [13], a neural network is proposed for ontology matching that expands the ability to represent and display complex relationships in a neural network for identical elements searching (Identical Elements Neural Network). The network can learn the high-level properties of concepts in various tasks and use them to find a correspondence between tasks. This allows to train the network to find correspondence for specific concepts between different problem areas.

The transformation of the task of mapping ontology concept instances into the problem of binary classification and its solution using machine learning methods is proposed in [14]. To reach this goal, a similarity vector was developed, independent of the presence of coinciding properties in the instances of concepts. The training of neural network to classify pairs of concepts for the sets of matches and mismatches is carried out based on this vector. To improve performance, it is suggested to use information about already existing alignments. As a classification mechanism, a random decision tree was chosen to construct weak classifiers, which are combined using the AdaBoost algorithm.

In this paper, it is proposed to use the neural network based on the TensorFlow framework implementing the Word2Vec model for comparing ontology concepts. Previously, this model was used in similar tasks, in particular, to build a common taxonomy of several [18,21], for research ontologies in different languages [5]. The original is the additional training of a pre-trained model to take into account the peculiarities of the semantic links between the names of

concepts in specific ontologies in order to more accurately match the semantics of the ontology problem domain.

The paper is structured as follows. Section 2 briefly describes Word2Vec model and how it can be used for ontology matching. Section 3 provides structure of neural network to implement Word2Vec model in terms of neural networks. In Sect. 4 the ontology matching method based on the Word2Vec neural network is described. Section 5 provides implementation of proposed method and Sect. 6 describes some experiment results.

2 Word2Vec Model

As a source of background knowledge for ontologies matching, it is proposed to use the vector word space, reflecting the semantics of the problem domain. This approach is part of the natural language processing associated with distributional semantics.

The main idea of distributional semantics is the mapping of each word of text to a contextual vector of large dimension, the components of which are represented by real number. A set of vectors forms a vector space of words in which semantically close words will be located closer than words not related by semantics (see Fig. 1).

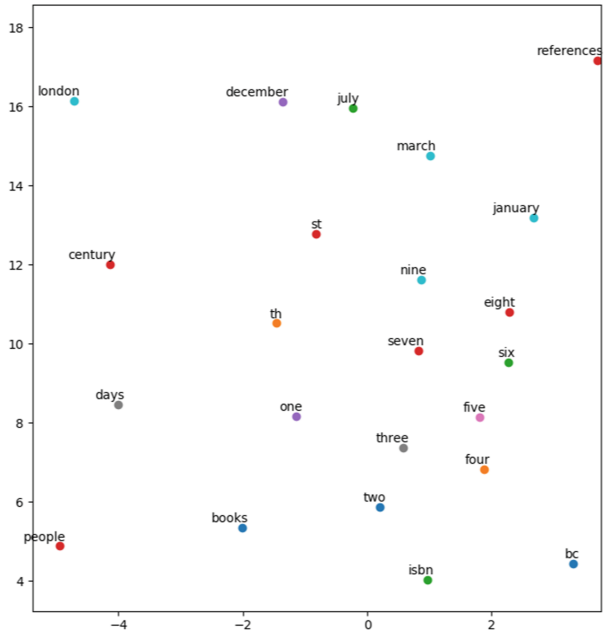


Fig. 1. Example of vector space

One of the main groups of models used to implement the vector representation of words is Word2Vec [10–12]. Its use allows both to predict a word depending on a context (continuous bag-of-words architecture), and to predict a context by a word (skip-gram architecture) using single vector model. Moreover, the word order in the context is not an important parameter (due to the so-called "bag of words" principle).

The vector space of words is represented by a matrix whose dimensions are determined by a dictionary size (the number of words that can be recognized) and the dimension of the vector. The greater the dimension of the vector used to display the word, the more accurately the semantic meanings of the words can be separated. To determine the parameters of the vector, training based on the analysis of the context of the word is used. For this, latent semantic analysis is used. [2,8], softmax using the Huffman tree (to reduce calculation) or the inverse problem of maximizing the objective likelihood function, minimizing log-likelihood of samples. During the training, the semantic proximity between words is calculated on the basis of their positions in the text and/or ontology graph and an assessment is made of the hidden factors that make up the components of word vector. In this case, semantically close words in the vector space will be located side by side, which allows the use of the vector space of words when comparing ontologies to define semantically close ontology concepts. The task of the neural network in this case is to find the closest concepts of compared ontologies in the vector space on the basis of a trained vector model of names of ontologies concepts after the learning process.

3 Neural Network Description

This section describes the neural network that implements the Word2Vec model and the basic principles on which the ontology concepts matching is built.

3.1 Structure

The purpose of using a neural network is to analyze ontology concepts and cross-search for similar concepts in matching ontologies. The neural network work is based on the vectorization of ontology concepts and the subsequent training of a neural network in the search for correspondences between new concepts and familiar ones. The basis is the group of statistical word2vec models and the corresponding neural network that implements these models.

The neural network can be divided into three layers: the input layer, the hidden layer and the output layer. The input layer corresponds to the dimension of the word vector. The hidden layer of the neural network stores a vector model of titles, and can be represented by a matrix of size $[m, n]$, where m is the number of words in the model dictionary, and n is the size of the word vector. The dimension of the output layer corresponds to the size of the dictionary. The neurons of the output layer contain the sigmoid activation function [3], which activates neurons depending on the value of the vector in hidden layer. Neurons

with the maximum value of the activation function will contain the names of the concepts that are as close as possible to the name of the ontology concept, which is input to the neural network. The result of the neural network are the values of similarity coefficients for each of the elements of the output layer.

3.2 Learning

Learning for a hidden layer of a neural network requires processing of a large corpora of natural language texts. Examples of such cases are encyclopedias articles, electronic libraries, and news or messages archives. An obvious advantage of this approach is control over the learning process, which consists in using only domain-specific texts. However, the learning process takes a significant amount of time (from 12 h when using one device with CPU and without graphics accelerators) or requires significant computational resources (with graphics accelerators, or parallelization of computations on several devices).

The learning process can be skipped when using a previously trained vector word model containing the semantics of the problem areas of the ontologies under consideration. Such a model will contain a large number of words, but due to the dimension of the vector space it will require very large volume and RAM size. The model used can be further trained on the context of ontology concepts (neighboring concepts, relationships between concepts, characteristics of concepts), so that it most fully reflects the semantics for a specific ontology. During training, the presence of ontology terms in the vector model is checked. If the word vector is present in the vector model, then for neighboring concepts and ontology properties, the elements of the corresponding vectors are adjusted to bring them together in vector space. Adjustment is carried out by the backward propagation method by calculating the difference between the expected output of the neural network and the real output, and applying the difference to the vectors of the hidden layer inside the neural network. In the case of the absence of a vector in the vocabulary, it is added to the model along with the corresponding output neuron, after which the elements are calculated for the new vector also on the basis of the back propagation of the error. The use of a pre-trained vector word model is necessary in order to exclude a situation in which there is no vector representation of the concept label from the compared ontology. If the label of the concept is not in the model, then it will be ignored, since the capabilities of the neural network are limited to the names included in the dictionary. One of the possible solutions to this problem is the stemming (search for the basis of the word) and lemmatization (bringing the word form to the normal (vocabulary) form) of words, followed by the search for the corresponding forms in the vector model. The trained vector model of words can also be saved for future use in order to reduce the learning time of the neural network for the ontology studied.

4 Ontology Matching Model

The general scheme of the developed method is based on expanding the ontology matching process developed by the project team earlier by adding background

knowledge hidden in the vector space of words. The input method accepts the ontologies O_1 and O_2 in the OWL format and returns the result of the comparison in the form of an alignment matrix for ontology concepts A as well as its visualization in the form of a bipartite graph. The basic scheme of the method is based on the use of entities labels and the context of the concepts of these ontologies.

The work of the method can be divided into three stages (see Fig. 2). The first stage is related to the processing of initial ontologies, during which the abstract and operational contexts are identified, as well as the problem area of each ontology is extracted. At the second stage, the mapping of processed ontologies based on methods using matching patterns and concept contexts takes place. To improve the quality of comparison methods different sources of background knowledge are used. The work of the matching methods at the second stage is performed in parallel and independently with the formation of intermediate matrices of ontology concepts. At the third stage of work, the composition of intermediate correspondence matrices is carried out with the formation of the final matrix, which is the final result of the work of the developed ontology matching method.

Consider in detail each of the stages. The first stage is the processing of the initial ontologies. During the processing, abstract and operational contexts are allocated for each ontology $F(O_i) = (AK_i, OK_i)$, reflecting the structure and instances of ontology concepts, respectively. The abstract context is

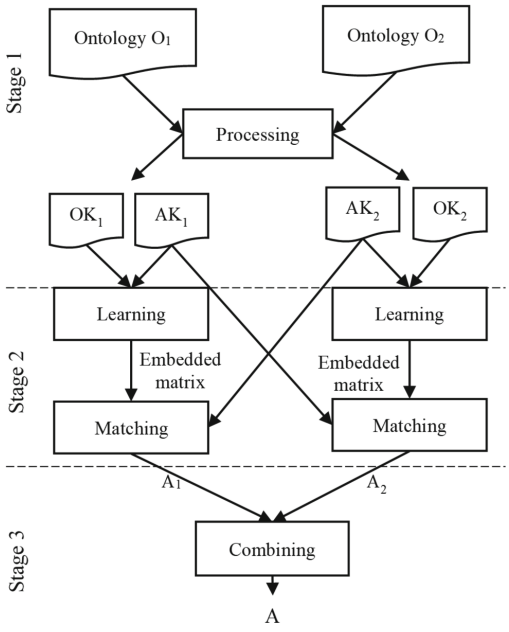


Fig. 2. The scheme of ontology matching method

represented using the object network of constraints [17], which allows to store more information about the structure of concepts and the ontology itself, which is subsequently used in the search for matching concepts. The operational context includes instances of ontology concepts, taking into account the values of their properties, the type and nature of links. For the pair of ontologies, the problem area is determined by the operational context using topic modeling methods [19]. Since these methods are used primarily to build topic models of texts, ontologies need to be converted into a textual representation, for which information from the operational context is used. The texts in this case are all the labels of the concepts of ontologies and the text parameters of the entity instances, collected in the so-called "bag of words".

The second stage is the comparison. Due to the fact that the proposed method implies cross-matching of ontologies, at the second stage two sub-stages can be distinguished, associated with the matching of each ontologies.

When searching for correspondence between concepts, the matching ontologies O_1 and O_2 are transmitted in the processed format of abstract and operational contexts to the neural network. From each ontology, the string characteristics of the concepts are retrieved - the labels of the concepts, their properties, relationships. A pre-trained vector word model is loaded into the hidden layer of the neural network. Then the neural network is additionally trained using first O_1 ontology concepts to recognize their belonging to a given ontology. The training based on the fact that for each ontology concept its semantic proximity with neighboring concepts and links with them is checked. If the value of the output neurons does not correspond to the context of the concept, the corresponding vectors are recalculated. Upon completion of the learning process, concepts from the O_2 ontology are fed to the input of the neural network and are checked to which concepts from the O_1 ontology they most fully correspond (the value of the activation function is checked, which lies in the range $[0, 1]$, where 0 is the lack of correspondence, 1 is full compliance). The maximum value of the activation function and its corresponding concepts are recorded in the intermediate matrix of ontology correspondence A_1. After checking all the concepts from the ontology O_2 in a similar way, the ontologies are replaced - the network is retrained to recognize the concepts from the ontology O_2 and checks the consistency of the concepts from the ontology O_1.

At the third stage, the intermediate comparison results are combined in the final alignment matrix by calculating the average value of each coefficient $A = 1/2A_1 + 1/2A_2$. The choice of simple arithmetic average is due to the equivalence of matching procedures.

5 Ontology Matching Model Implementation

To implement the Word2Vec model in the neural network, the TensorFlow library was used. The training was based on English Wikipedia texts (dump from 2017, 4 billion words with a vector space dimension of 300 - the size of the trained model is about 8 GB), which were pre-processed to eliminate service words, unions and

words that are not part of the problem area. The last is required in order to reduce the dimension of the embedding matrix during training and use. Also used pre-trained model on the news texts from Google News. The size of the pre-trained model dictionary is about 3 million words and phrases, and the vector space dimension is 300 components (the file size of the trained model requires about 3 GB of HDD space to store).

Python v3.7 was chosen as the main programming language for the development of the prototype. The main advantage of using it for the presented task is the availability of powerful modules for processing science data with large data arrays, such as numpy, scikit, etc. which have a simple interface and C kernel implementation, as well as the TensorFlow library for configuring and evaluation a neural network.

Ontologies in OWL are loaded and processed using the owlready2 library. This library allows to convert ontology classes into Python objects of the and vice versa, which greatly simplifies their processing.

For the convenience of displaying the found matches, visualization based on a weighted bipartite graph was used. The vertices of the graph are the concepts of ontologies, arcs are correspondence between concepts and the weights of the arcs are the coefficient of conformity of the concepts. The networkx library was used for visualization.

The software architecture of the ontology matching module is presented in the Fig. 3.

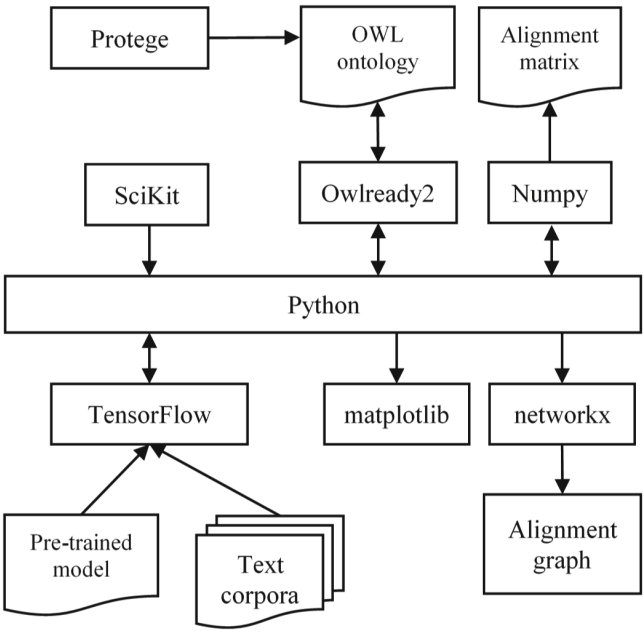


Fig. 3. Software architecture of ontology matching module

6 Evaluation

To test the proposed approach, a set of data used in the competition on ontology comparison - OAEI was used. Each year, several data tracks are formed from various problem areas in which developers offer their methods for comparison and determine the best estimated precision, recall and F-Measure. The proposed approach was tested on the Conference track data set of 2018.

Conference track contains 16 ontologies united by the domain of conference organization. These ontologies are developed to be suitable for ontology matching task because of their heterogeneous character of origin (see Fig. 4).

Ontologies have been based upon three types of underlying resources:

- actual conference and its web pages,
- actual software tool for conference organization support,
- experience of people with personal participation in organization of actual conference.

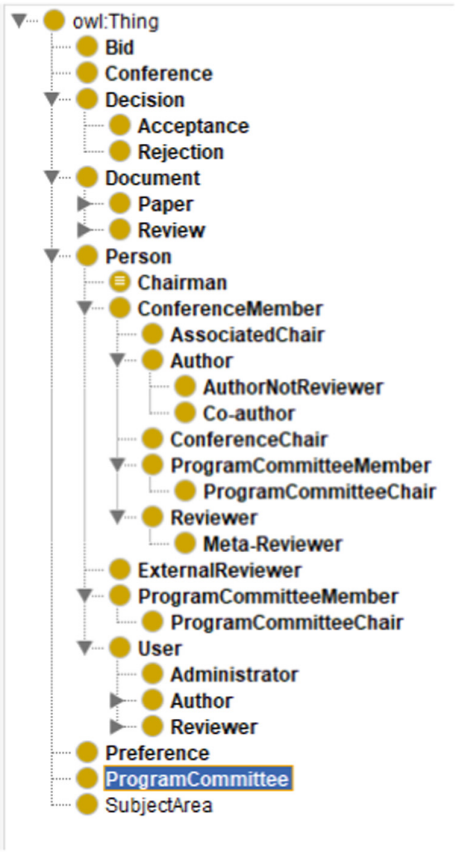


Fig. 4. An example of an ontology from the conference suite

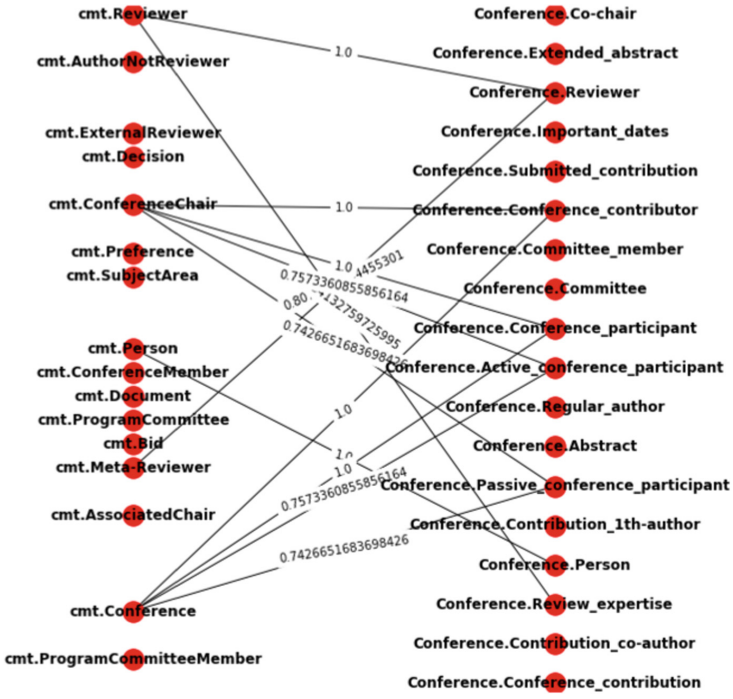


Fig. 5. The result of comparing two ontologies

Table 1. Evaluation metrics

Metric	Proposed approach	ALOD2Vec	StringEquiv	AML
Precision	0.79	0.71	0.8	0.84
Recall	0.65	0.5	0.43	0.66
F1-measure	0.36	0.59	0.56	0.74

All experiments were performed on a test bench with the following configuration: CPU Intel(R)Core(TM)i7-4770K 3.1 GHz, RAM 16G, and GPU NVidia(R)GeForce 770. The matching of ontologies was carried out in pairs, $C_m^2 = 120$ matching procedures were made. Figure 5 presents examples of matching results for the random ontologies from the set.

According to the results of the proposed method tests, the accuracy of the work was evaluated. Traditionally, ontology comparison problems were estimated by parameters precision (the fraction of retrieved documents that are relevant to the query), recall (percent of all relevant documents that is returned by the search) and F-measure. The results are presented in Table 1 with comparison of other approaches from *ra1-M3* evaluation modality that is ontologies that can be downloaded and compared by using both classes and properties [22].

Low F-measure can be explained by the number of all ontology concepts exceeds the number of coinciding ones. In addition, the proposed method requires additional configuration in the part of the neural network learning process for ontology concepts to account for more background knowledge sources.

7 Conclusion

The results of the work show that the use of background knowledge, enclosed in the semantics of the problem area, really allows to process for the ontology matching task. The use of a neural network that implements the model of the vector space of words allowed to partially extract the background knowledge and utilize it to analyze the names of ontology concepts. The results obtained can be improved by analyzing the structure of the ontology: characteristics of concepts, instances, domains of characteristics, analysis of types of links.

As directions for further work, we can highlight the study of existing pretrained models and different implementations of word vector space on their completeness and accuracy of semantic separation, which is important when comparing ontologies from similar problem areas. Another focus is the creation of corpuses of texts and the training of models for frequent problem areas. The main goal in this case is to reduce the dimension of the embedding matrix, by eliminating concepts that are not correspond to used problem area without losing the semantic connectedness of the remaining concepts. In addition, reducing the size of buildings for training will speed up the learning process.

References

1. Chaves, M.S., de Freitas, L.A., Vieira, R.: Hontology: a multilingual ontology for the accommodation sector in the tourism industry. In: Filipe, J., Dietz, J.L.G. (eds.) KEOD, pp. 149–154. SciTePress (2012). http://dblp.uni-trier.de/db/conf/ic3k/keod2012.html#ChavesFV12
2. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. J. Am. Soc. Inform. Sci. **41**(6), 391–407 (1990)
3. Elfwing, S., Uchibe, E., Doya, K.: Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. Neural Netw. **107**, 3–11 (2018). https://doi.org/10.1016/j.neunet.2017.12.012
4. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38721-0
5. Gromann, D., Declerck, T.: Comparing pretrained multilingual word embeddings on an ontology alignment task. In: Proceedings of the 11th International Conference on Language Resources and Evaluation, pp. 230–236 (2018)
6. Hamdi, F., Reynaud, C., Safar, B.: Pattern-based mapping refinement. In: Cimiano, P., Pinto, H.S. (eds.) EKAW 2010. LNCS (LNAI), vol. 6317, pp. 1–15. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16438-5_1
7. Hecht, T., Buche, P., Dibie, J., Ibanescu, L., dos Santos, C.T.: Ontology alignment using web linked ontologies as background knowledge. In: Guillet, F., Pinaud, B., Venturini, G. (eds.) Advances in Knowledge Discovery and Management. SCI, vol. 665, pp. 207–227. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-45763-5_11

8. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse Process. **25**(2–3), 259–284 (1998). https://doi.org/10.1080/01638539809545028
9. Lin, F., Sandkuhl, K.: A survey of exploiting WordNet in ontology matching. In: Bramer, M. (ed.) IFIP AI 2008. ITIFIP, vol. 276, pp. 341–350. Springer, Boston, MA (2008). https://doi.org/10.1007/978-0-387-09695-7_33
10. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space, pp. 1–12, January 2013. http://arxiv.org/abs/1301.3781
11. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality, pp. 319–333, October 2013
12. Onal, K.D., et al.: Neural information retrieval: at the end of the early years. Inform. Retrieval J. **21**(2–3), 111–182 (2018). https://doi.org/10.1007/s10791-017-9321-y
13. Peng, Y., Munro, P., Mao, M.: Ontology mapping neural network: an approach to learning and inferring correspondences among ontologies. In: CEUR Workshop Proceedings, vol. 658, pp. 65–68 (2010)
14. Rong, S., Niu, X., Xiang, E.W., Wang, H., Yang, Q., Yu, Y.: A machine learning approach for instance matching based on similarity metrics. In: Cudré-Mauroux, P., et al. (eds.) ISWC 2012. LNCS, vol. 7649, pp. 460–475. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35176-1_29
15. Scharffe, F., Euzenat, J., Fensel, D.: Towards design patterns for ontology alignment. In: Proceedings of the 2008 ACM Symposium on Applied Computing - SAC 2008, p. 2321. ACM Press, New York (2008). https://doi.org/10.1145/1363686.1364236
16. Scharffe, F., Zamazal, O., Fensel, D.: Ontology alignment design patterns. Knowl. Inform. Syst. **4**(1), 1–28 (2014). https://doi.org/10.1007/s10115-013-0633-y
17. Smirnov, A., Levashova, T., Kashevnik, A.: Ontology-based resource interoperability in socio-cyber-physical systems collaboration scenario. Inform. Technol. Ind. **6**(2), 19–24 (2018)
18. St. Chifu, E., Letia, I.A.: A neural model for unsupervised taxonomy enrichment. In: Proceedings of the 10th International Conference on Information Integration and Web-based Applications and Services, iiWAS 2008, pp. 264–270 (2008). https://doi.org/10.1145/1497308.1497358
19. Steyvers, M., Griffiths, T.: Probabilistic topic models. Handb. Latent Semant. Anal. **427**(7), 424–440 (2007)
20. Svab-Zamazal, O., Svatek, V.: Towards ontology matching via pattern-based detection of semantic structures in owl ontologies. In: Proceedings of the Znalosti Czecho-Slovak Knowledge Technology conference (2009)
21. Wohlgenannt, G., Minic, F.: Using Word2Vec to build a simple ontology learning system. In: CEUR Workshop Proceedings, vol. 1690, pp. 2–5 (2016)
22. Zamazal, O., Cheatham, M., Jana, V.: Results of evaluation for the conference track within OAEI 2018 (2018). http://oaei.ontologymatching.org/2018/results/conference/index.html#ra1-M3. Accessed 5 May 2019