An Eye on Representation Learning in Ontology Matching

Guilherme Sousa[1,2], Rinaldo Lima[2][0000−0002−1388−4824], Cassia Trojahn[1][0000−0003−2840−005X]

[1] Institut de Recherche en Informatique de Toulouse, Toulouse, France
guilherme.santos-sousa@irit.fr, cassia.trojahn@irit.fr
[2] Universidade Rural de Pernambuco, Recife, Brazil
rinaldo.jose@ufrpe.br

Abstract. Representation learning has received increased attention in the last few years in several tasks, including knowledge graph completion, entity resolution, and ontology matching. This paper presents an overview of representation learning approaches applied to the ontology matching task. It proposes to classify such approaches to the following dimensions: lexical unit segmentation, training strategy, and information representation complexity. A discussion on them is presented together with their pros and cons. Perspectives for further developments are also discussed.

1 Introduction

The advances in machine learning have promoted the emergence of new architectures and methods capable of learning complex patterns on different types of data. Some applications show impressive results on NLP tasks, such as question answering and summarisation [3], image generation [41], or still in general tasks such as game playing and image captioning [42]. In knowledge representation, representation learning methods show significant results in tasks such as graph completion [9] and link prediction [22]. In ontology matching, a wave of representation learning systems has appeared in the last few years. Word embeddings were one of the first representation learning strategies adopted [58]. These models are based on the distributional hypotheses stating that similar words appear in similar contexts. The well-known Word2Vec [15] model has been used to compute the semantic similarity of ontology entities that can improve systems performance compared to classical lexical similarity methods [31].

Since word embeddings do not consider the ontology structure, better encoding strategies were designed. The RDF2Vec[43] is used to integrate background knowledge in ontology matching systems such as ALOD2Vec [39] and [38]. It generates sentences by randomly walking the paths in ontology and uses the Word2Vec algorithm to generate entity embeddings. Besides its representation capabilities, RDF2Vec does not fully represent OWL constraints, without considering as well the word composition in labels, harming the model's generality. To address these problems, the OWL2Vec model [6] was designed by using a set of rules to map OWL predicates to RDF equivalents and considering the word composition of entity labels. Since these models generate

independent embeddings for entities in two different ontologies, post-processing strategies like the alignment of embeddings using linear transformations [38] or Siamese networks [7] were used. Besides the performance of random walk-based strategies, as adopted in RDF2Vec and OWL2Vec, Graph Neural Networks [55] are more flexible and show impressive performance in generating graph embeddings [5, 53]. The Graph Attention Network [49], for instance, can filter irrelevant neighbors in entity encoding leading to more robust similarity metrics.

For better representation of terminological layers, Transformers [48] and BERT [11] language models were introduced in ontology matching systems [16, 25, 28]. They have achieved high performance and capacity of learning language representations in NLP tasks [40]. They have attracted interest in generating embeddings for entity labels. BERT is a language model based on the Transformer architecture [48] and has outperformed Recurrent Neural Networks as they rely on an attention mechanism to filter irrelevant data. BERT enables the architecture development of huge models with billions of parameters [3]. It is pre-trained by masking some input tokens and by predicting the next sentence. This task provides the resulting model with higher context notion capabilities in different NLP tasks by fine-tuning the model for each task. In ontology matching, transformer-based models are used to learn representations for the ontology terminological layer (labels, comments, etc).

The goal of this paper is to review ontology matching systems based on representation learning. These systems have some properties in common that enable them to be categorized based on the following dimensions: lexical unit segmentation, training, and information complexity. A discussion on the performance of variant systems to ontology matching is presented together with their pros and cons. The perspectives for further developments are also discussed. We start by presenting a categorization framework (§2), followed by the description of the reviewed proposals (§3). We then discuss their impact on ontology matching (§4) together with an analysis for further work (§5).

2 Categorisation framework

The works in the literature are categorized into three groups of common features: lexical unit segmentation, training, and information complexity. In the next sections, these groups are discussed in detail. The reviewed works are listed in Table 1.

2.1 Lexical unit segmentation

The lexical unit segmentation categorizes the systems in the way they represent the terminological layer of the input ontologies: entity (no tokenization), words, and character. An example of the categories is illustrated in Figure 1. **Entity level** refers to the systems where no tokenization is applied in the terminological layer (id, labels, comments, etc), viewing these terminological features as a unique symbol (ex. ”New_York_City”). It acts as an identifier and has low generalization since ontologies may combine the same words in a different order to represent the same entity. **Word level** refers to the splitting of the terminological features in their word components (ex. [”New”, ”York”, ”City”]. In order to generate embeddings with the same dimension, an aggregation step needs

	Lexical Unit			Learning			Information level			
Work	Entity	Word	Char	Ref.	Trained	Pre-trained	No Context	Context	Graph	BK
Zhang [58]		✓				✓	✓			
Xiang [56]		✓			✓			✓		
Kolyvakis [26]		✓			✓	✓				✓
Kolyvakis [27]		✓			✓	✓				✓
Nkisi [37]		✓		✓	✓	✓		✓		
Gromann [14]		✓				✓	✓			
Jimenez [23]		✓			✓		✓			
OntoEmma [50]		✓	✓	✓	✓	✓				✓
Li [32]		✓			✓			✓		✓
HISDOM [33]		✓				✓		✓		
Tounsi [47]		✓				✓		✓		
DOME [18]		✓			✓			✓		
Li [31]		✓				✓	✓			
DeepFCA [30]		✓	✓		✓	✓		✓		✓
Bento [2]			✓	✓	✓	✓		✓		
ALOD2Vec [39]	✓				✓					✓
LogMap* [7]	✓	✓			✓				✓	
DAEOM [53]		✓		✓	✓	✓			✓	
BERTMap [16]		✓			✓	✓				✓
OntoConnect [5]			✓		✓	✓			✓	
VeeAlign [21]		✓		✓	✓	✓			✓	
AMD [52]	✓				✓				✓	
Fine-TOM [25]		✓		✓	✓	✓	✓			

Table 1. The reviewed works are categorized according to Lexical Unit Segmentation, Training, and Information level, sorted by publication year.

to be applied like summing or averaging the embeddings of each word. Most works in ontology matching use this approach since many pre-trained word embeddings are available and the input ontologies in the same domain share overlapping vocabulary. **Character level** is more generalist as it can be applied to any domain with the same alphabet. This type of approach requires a complex model to generate relevant representations as word boundaries need to be learned by the combination of characters.

2.2 Learning

Some works are supervised and need labeled data for training while others are based on pre-trained models or still on unsupervised learning. **Reference** refers to those systems that need reference alignments for training the model. The **Training** category classifies those systems that need to fine-tune or learn weights for each ontology before the production of final alignments. **Pre-trained** refers to the systems that contain background knowledge in the form of pre-trained word embeddings or models.

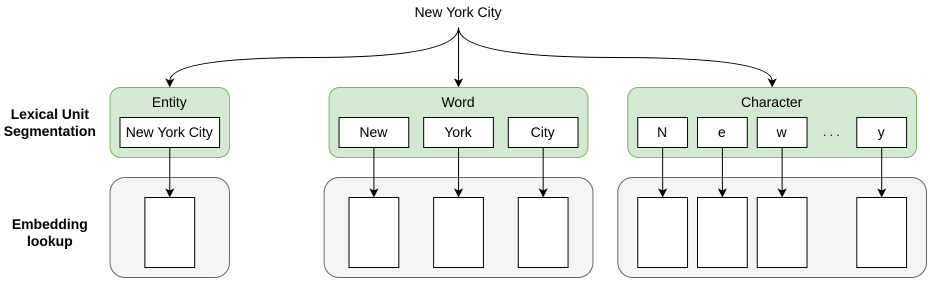

Fig. 1. An example of the label "New York City" segmented in the Entity, Word, and Character levels. Optional lower casing can be applied.

2.3 Information complexity

This group describes how much information is considered in the matching process. The **No Context** category describes the systems that do not consider the entity neighborhood. **Context** characterizes the systems that consider neighborhood information without considering the graph structure (e.g., predicates or graph edge directions). The **Graph** category groups the architectures that consider the neighbors and edge features of the relations that connect them in arbitrary depth. The **Background Knowledge** category groups the systems that can aggregate information that is not present in the ontology (e.g., external dictionaries or thesauruses).

3 Works on the Information Complexity Category

We chose to describe the works that are grouped according to the category *information level*, which is the most discriminant one. This category classifies the works according to four main features (Table 1): i) considering element-level based strategies (no context); ii) systems exploiting context; iii) system using deeper context (graphs); and iv) systems using external background knowledge.

3.1 Element-level based systems (no context)

This category groups the works that map entity's terminological layer to embeddings without considering it's neighborhood. The work of [58] is claimed to be the first work in ontology matching that used word embeddings, which were learned on Wikipedia data using two techniques: Word2Vec [36] and Latent Semantic Analysis [12]. Another relevant work using word embeddings is presented in [31]. The entity label is tokenized and each token is mapped to a word embedding from a biomedical domain corpus. Despite its simplicity, this method improved lexical similarities like edit distance. While these works have focused on monolingual ontologies, the system in [14] uses pre-trained word embeddings to align multilingual ontologies. A cross similarity between each word embedding, corresponding to the compared entity labels, is used to build an intermediate vector to produce the final similarity. To deal with the *Out Of*

Vocabulary (OOV) problem, this work uses the average of the similarity of the present words.

While few works expose their implementation as modules or libraries, the MELT platform [19] has support for the use of pre-trained transformers in a pipeline for ontology matching. FineTOM [25, 28] is an system based on MELT and that uses transformers and two pipelines. The first step is a training pipeline to fine-tune a pre-trained transformer and the second is a matching pipeline that uses the transformer to measure similarity between entities. Despite the capacity of transformer-based models to generate contextualized word embeddings, the TOM and Fine TOM were classified in No Context category as these systems don't consider information from the entity neighborhood in the entity embedding.

Finally, with a focus on large ontologies, the work in [23] proposes a technique to cluster entities using an inverse word index to reduce the required matching space. Two clustering strategies are applied to generate the subtasks based on the inverse word index. The first is the random splitting of the entities in clusters of the same size and the second is the use of the StarSpace [54] embedding model to cluster related entities while learning embeddings for individual words.

3.2 Systems exploiting context

This category refers to the works that include information about the surroundings of each entity, without exploring the full graph structure. The work of [37] uses a combination of automatically learned and manually engineered features containing word embedding similarities. Four similarity metrics are proposed, combining context and entity labels similarity that are used to build a feature vector. The final alignment is produced using a random forest classifier to predict the alignment using the generated features. In [47], word embeddings are used to represent the entities as the average of the embeddings related to entity labels words and its context as the average of the embeddings in the entity lineage. The final alignment is generated by comparing the cosine similarity between the entity and context embeddings on the source and target ontologies. The system can infer the specific relation between the alignment using the radius measure of their contexts. Similar work, but using character level, is proposed in [2]. The entity embedding is generated using a CNN on the characters of the entity labels, its parents, and children. The character embedding is pre-trained and the last layer of the CNN is connected to a single neuron with sigmoid activation to predict the alignment probability.

The architecture ERSOM (Entity Representation and Structure-based Ontology Matching) [56] uses a stacked auto-encoder to learn features from entities based on different methods to encode classes, properties, and instances. Each entity representation aggregates context information based on its surroundings and hidden representations are learned using the auto-encoder. In the same sense, HISDOM [33] uses different similarity metrics for instances, names, attributes, structure, and comments. The name similarity is a weighted sum between edit distance and embedding cosine similarity in the entity labels. In particular, the structural similarity is calculated as the weighted sum of the Jaccard similarity of the children and parent concepts, with the comment similarity being calculated embedding the sentence of the entity comments using a CNN. The

final similarity is the weighted sum of all similarities. Finally, DOME [18] applies a pipeline of filters. The first one is a string similarity matcher that compares the textual description of entities. The next step is a confidence adjustment using an embedding generated by Doc2vec [29], a generalization from Word2vec, to calculate cosine similarity. After instance and property alignment were performed, the class alignment is revisited using the matched instances as context to the class similarity since classes of matched individuals tend to be the same.

3.3 Graph as context

Graph-based category refers to the systems representing the ontology graph structure in depth. The OWL2Vec [6] model is similar to RDF2Vec[43], however, it differs in the lexical unit segmentation as OWL2Vec combines entity and word representation while RDF2vec only considers entity lexical units. LogMap* system [7] uses this embedding model, and since the generated embeddings are independent, a siamese network is used to learn a transformation that projects the embeddings into the same space.

OntoConnect [5] uses a recursive neural network to encode the graph structure of the entities in an unsupervised manner. The entity features are generated using FastText [35], an embedding model developed by Facebook based on character n-grams. A recursive neural network based on LSTM [57] is used to embed the graph structure of each entity. Also using graph neural networks DAEOM [53] (Deep Attentional Embedded Ontology Matching) divides its architecture into Ontology Attentional Encoder and Mapping Selection. The first step of the Encoder is the embedding of the terminological descriptions. A fine-tuned architecture based on BERT [11] is used to embed the words of textual data in the entities. Next, a Graph Attention Network (GAT) is used to aggregate the graph structure of the entity with its terminological layer embedding. Finally, VeeAlign [21] is a system that builds entity embeddings using the Universal Sentence Encoder [4] and aggregates four views to generate the context embedding: parents, child, properties, and datatype properties. A path and node attention is applied to select the most important representations. The entity embedding is then concatenated to the context embedding and passed through a feed-forward layer for down-sampling.

An example of the use of translational embedding to ontology matching [22] is AMD (AgreementMakerDeep) [52]. This type of embeddings was shown to be able to model some logical relations like inversions, symmetries, and some compositions that appear in ontologies entities (ex. Inverse Properties). It uses a modified version of RotatE [46] to generate embeddings of the ontologies and compare similarities.

3.4 Background knowledge

This category is dedicated to the works that use external knowledge in the matching process. Most systems use pre-trained embeddings and models as background knowledge as Alod2Vec [39] which uses embeddings encoded with RDF2Vec [43] on an external dataset. Other systems like DeepAlignment [26] refine word embeddings using synonyms to increase their semantic similarity. Pre-trained word embeddings are refined by contrasting synonym samples from background knowledge and negative samples from entities that are not explicitly stated as equivalent. BERTMap [16, 17] system

also extract synonyms from background ontologies to fine-tune a BERT architecture. It also includes a mapping repair to increase the quality of final alignments. A different strategy is adopted in [27], where a Siamese CBOW [24] refines the word embeddings by extracting paraphrases from background knowledge and a Denoising Auto Encoder (DAE) [34] to learn entity embeddings.

In DeepFCA [30] word embeddings and formal concept analysis [13] are adopted. A pre-trained word embedding is used to map the words in the terminological layer using a character-level embedding as a fallback if the word is not present. The entity representation is created by averaging the word vectors refined by synonyms extracted from semantic lexicons and contrasting them with negative samples from the ontology, guided by a lattice generated from formal concept analysis.

The system in [32] uses multi-view embedding, a strategy that combines embeddings from different views as parent and children ontology concepts, to calculate similarities between entities. Embeddings are learned using a negative sampling strategy to contrast entity synonyms extracted from external ontology with random samplings. Finally, OntoEmma [50], as in the previous work, combines representations from different views to generate entity embeddings. The first is the name view that concatenates character-level embeddings with pre-trained word embeddings and is aggregated using a bidirectional LSTM. The aliases, definition, and context views use the same strategy as the name view without the use of character embeddings and are enriched with background knowledge. The final alignment similarity is calculated by a neural network to predict the alignment probability.

4 Discussion

As introduced above, many representation learning strategies have been applied to ontology matching. Most of the works shown in Table 1 are based on word lexical segmentation and trained per ontology. They use pre-trained word embeddings and, some of them, rely on pre-trained models (e.g BERT), which takes into account some types of contextual information. A qualitative analysis of the selected OM systems shows that not only good capacity for generalization, but also higher level of flexibility can be achieved when representation learning methods were applied. In the remainder of this section, we point out common difficulties found when applying representation learning methods in ontology matching. In addition, some further developments to address such difficulties are also discussed.

Lexical Unit Segmentation The use of representation learning in ontology matching has some common problems that need to be considered. One problem in this dimension is that the non-determinism of neural learning-based methods and the leads to encoding of two different ontologies that do not share vocabulary may lead to produce meaningless similarity scores [7]. Some works [7, 38] propose solutions to this problem by projecting the embeddings of the mapped ontology into the same space using linear transformations or siamese deep learning networks. The choice of entity lexical unit segmentation, as described in Section 2.1, emphasizes this problem since similar entity labels with the same words, but in a different order, may lead to different representations with a low similarity between them. As an attempt to solve this problem,

word and character lexical unit segmentation can be applied [44, 10]. Moreover, word-level lexical units are also prone to the problem of out of vocabulary, when words not present in the training vocabulary harm representation robustness. The use of character embeddings to generate entity representations may be an interesting research area to mitigate this problem when combined with other representations. The performance of word embeddings in ontology matching is limited by the vocabulary coverage of the used embeddings restricting matching systems developers to use domain-specific word embeddings. Since the alphabet size of languages is orders of magnitude lower than the vocabulary size and different languages share an alphabet, the use of characters embedding can be used to reduce the occurrence of OOV at the cost of bigger models [51]. There is still space for research in the use of character embeddings for ontology matching that can improve the performance of general systems that works for different ontology domains and different languages.

Learning In the choice of learning strategy, one difficulty of applying supervised representation learning techniques is the low amount of reference alignments present in matching datasets [26], making unsupervised and self-supervised learning valuable techniques to model ontology concepts. However, the major challenge when employing such strategies consists in modeling a direct loss metric that reflects the similarity between two given entities. The general notion of similarity is in focus of ongoing research and can change among ontology domains. Concerning the training model dimension, recent works have demonstrated that relying only on word embeddings is not enough to fully address the matching problem. By the way, to reject false positives, the contextual information derived from graph structures in ontologies needs to be taken carefully into consideration. Actually, some graph embedding techniques achieve promising results using supervised learning techniques [21, 53]. However, due to the problem of a low amount of labeled data, such embedding techniques may not achieve their highest performance in ontology matching without some adaptation [59]. One possible direction to deal with the low amount of labeled data relies on unsupervised graph embedding methods, including translational models [22], AMD [52]. Unsupervised graph embedding methods have shown effective performance in encoding graph structures, especially for the Link Prediction task in Knowledge Graphs. These models have the advantage of being able to learn logical relations between entities, including inversion, symmetrical, and some types of relation compositions obeying the locality principle [45] in ontology matching. However, only relying on this type of embedding method does not ensure acceptable performance, since it does not enforce high similarity scores between similar entities in two distinct ontologies.

Information Level The use of embeddings brings the possibility to compare symbolic data similarities, by analogy, using metrics such as euclidean distance or cosine similarity. For designing metrics fully adapted to the ontology matching problem, the embeddings must be organized in a latent space enforcing that similar concepts have representations positioned near to each other in the same vector space. However, some models, e.g vanilla Auto Encoders do not enforce this type of organization in latent space. In this way, the learned embeddings of similar entities are not guaranteed. Some improvements in generative models such as Infogan [8], Variational Auto Encoder (VAE) [1]

and Beta-VAE [20] can learn expressive interpretable representations and can encode background knowledge information in it's models. They encode specific features in each embedding dimension making similar embeddings to be placed near each other in latent space. In ontology matching, these different dimensions could offer better generalization and interpretability of concepts. An interesting research direction is the use of these representation methods for encoding ontology entities, leading to embedding representations that have similarity metrics more suitable to the matching task. Last but not at least, all the works still focus on simple alignments. A further direction is to exploit such existing representation learning techniques to the task of generating expressive alignments. While many systems rely on context and the ontology graph structure, in complex alignment the notion of neighborhood is dissolved since multiple entities may represent one concept between ontologies. This may cause the necessity of change in existing architectures to deal with such a task.

5 Conclusion and Future Work

This paper presented an overview of ontology matching systems from the perspective of representation learning. The aforementioned systems were classified according to three categories revealing the mos relevant aspects of them. Our analysis pointed out that not only good capacity for generalization, but also higher level of flexibility can be achieved when representation learning methods were applied. However, there is still room for improvement as the selected works still do not fully explore all the possibilities that representation learning can provide. In this direction, new approaches should explore deep learning model with higher parameter dimensions as well as the integration of background knowledge and models composition.

References

1. J. An and S. Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1):1–18, 2015.
2. A. Bento, A. Zouaq, and M. Gagnon. Ontology matching using convolutional neural networks. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5648–5653, 2020.
3. T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
4. D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.
5. J. Chakraborty, S. K. Bansal, L. Virgili, K. Konar, and B. Yaman. Ontoconnect: Unsupervised ontology alignment with recursive neural network. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, pages 1874–1882, 2021.
6. J. Chen, P. Hu, E. Jimenez-Ruiz, O. M. Holter, D. Antonyrajah, and I. Horrocks. Owl2vec*: Embedding of owl ontologies. *Machine Learning*, 110(7):1813–1845, 2021.
7. J. Chen, E. Jiménez-Ruiz, I. Horrocks, D. Antonyrajah, A. Hadian, and J. Lee. Augmenting ontology alignment by semantic embedding and distant supervision. In R. Verborgh,

K. Hose, H. Paulheim, P. Champin, M. Maleshkova, Ó. Corcho, P. Ristoski, and M. Alam, editors, *The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings*, volume 12731 of *Lecture Notes in Computer Science*, pages 392–408. Springer, 2021.

8. X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in Neural Information Processing Systems*, 29, 2016.
9. Z. Chen, Y. Wang, B. Zhao, J. Cheng, X. Zhao, and Z. Duan. Knowledge graph completion: A review. *IEEE Access*, 8:192435–192456, 2020.
10. A. Conneau, H. Schwenk, L. Barrault, and Y. LeCun. Very deep convolutional networks for text classification. In *Proceedings of the 15th EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 1107–1116, 2017.
11. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
12. S. T. Dumais et al. Latent semantic analysis. *Annu. Rev. Inf. Sci. Technol.*, 38(1):188–230, 2004.
13. B. Ganter and R. Wille. *Formal concept analysis: mathematical foundations*. Springer Science & Business Media, 2012.
14. D. Gromann and T. Declerck. Comparing pretrained multilingual word embeddings on an ontology alignment task. In *Proceedings of the eleventh international conference on language resources and evaluation (LREC)*, 2018.
15. L. Gutiérrez and B. Keith. A systematic literature review on word embeddings. In *International Conference on Software Process Improvement*, pages 132–141. Springer, 2018.
16. Y. He, J. Chen, D. Antonyrajah, and I. Horrocks. Bertmap: A bert-based ontology alignment system. *arXiv preprint arXiv:2112.02682*, 2021.
17. Y. He, J. Chen, D. Antonyrajah, and I. Horrocks. Biomedical ontology alignment with BERT. In *Proceedings of the 16th Workshop on Ontology Matching*, pages 1–12, 2021.
18. S. Hertling and H. Paulheim. DOME results for OAEI 2019. In *Proceedings of the 14th Workshop on Ontology Matching*, pages 123–130, 2019.
19. S. Hertling, J. Portisch, and H. Paulheim. Matching with Transformers in MELT. *arXiv preprint arXiv:2109.07401*, 2021.
20. I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.
21. V. Iyer, A. Agarwal, and H. Kumar. Veealign: Multifaceted context representation using dual attention for ontology alignment. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10780–10792, 2021.
22. S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514, 2021.
23. E. Jiménez-Ruiz, A. Agibetov, M. Samwald, and V. Cross. Breaking-down the ontology alignment task with a lexical index and neural embeddings. *arXiv preprint arXiv:1805.12402*, 2018.
24. T. Kenter, A. Borisov, and M. De Rijke. Siamese cbow: Optimizing word embeddings for sentence representations. *arXiv preprint arXiv:1606.04640*, 2016.
25. L. Knorr and J. Portisch. Fine-TOM matcher results for OAEI 2021. In *Proceedings of the 16th Workshop on Ontology Matching*, volume 3063, pages 144–151, 2022.
26. P. Kolyvakis, A. Kalousis, and D. Kiritsis. Deepalignment: Unsupervised ontology matching with refined word vectors. In *Proceedings of the 2018 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 787–798, 2018.
27. P. Kolyvakis, A. Kalousis, B. Smith, and D. Kiritsis. Biomedical ontology alignment: an approach based on representation learning. *Journal of Bio. Semantics*, 9(1):1–20, 2018.
28. D. Kossack, N. Borg, L. Knorr, and J. Portisch. TOM matcher results for OAEI 2021. In *Proceedings of the 16th Workshop on Ontology Matching*, pages 144–151, 2021.
29. J. H. Lau and T. Baldwin. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368*, 2016.
30. G. Li. Deepfca: Matching biomedical ontologies using formal concept analysis embedding techniques. In *Proceedings of the 4th International Conference on Medical and Health Informatics*, pages 259–265, 2020.
31. G. Li. Improving biomedical ontology matching using domain-specific word embeddings. In *Proceedings of the 4th International Conference on Computer Science and Application Engineering*, pages 1–5, 2020.
32. W. Li, X. Duan, M. Wang, X. Zhang, and G. Qi. Multi-view embedding for biomedical ontology matching. *Proceedings of the 14th Workshop on Ontology Matching*, 2536:13–24, 2019.
33. J. Liu, Y. Tang, and X. Xu. HISDOM: A Hybrid Ontology Mapping System based on Convolutional Neural Network and Dynamic Weight. In *Proceedings of the 6th IEEE/ACM International Conf. on Big Data Computing, Applications and Tech.*, pages 67–70, 2019.
34. X. Lu, Y. Tsao, S. Matsuda, and C. Hori. Speech enhancement based on deep denoising autoencoder. In *Interspeech*, volume 2013, pages 436–440, 2013.
35. T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
36. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.
37. I. Nkisi-Orji, N. Wiratunga, S. Massie, K.-Y. Hui, and R. Heaven. Ontology alignment based on word embedding and random forest classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 557–572. Springer, 2018.
38. J. Portisch, G. Costa, K. Stefani, K. Kreplin, M. Hladik, and H. Paulheim. Ontology matching through absolute orientation of embedding spaces. *arXiv preprint arXiv:2204.04040*, 2022.
39. J. Portisch, M. Hladik, and H. Paulheim. ALOD2Vec matcher results for OAEI 2020. In *Proceedings of the 15th Workshop on Ontology Matching*, pages 147–153, 2020.
40. X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang. Pre-trained models for natural language processing: A survey. *CoRR*, abs/2003.08271, 2020.
41. A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with CLIP latents, 2022.
42. S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez, Y. Sulsky, J. Kay, J. T. Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.
43. P. Ristoski, J. Rosati, T. Di Noia, R. De Leone, and H. Paulheim. RDF2Vec: RDF graph embeddings and their applications. *Semantic Web*, 10(4):721–752, 2019.
44. R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016.
45. A. Solimando, E. Jiménez-Ruiz, and G. Guerrini. Detecting and correcting conservativity principle violations in ontology-to-ontology mappings. In *International Semantic Web Conference*, pages 1–16. Springer, 2014.

46. Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.
47. M. Tounsi Dhouib, C. Faron Zucker, and A. G. Tettamanzi. An ontology alignment approach combining word embedding and the radius measure. In *International Conference on Semantic Systems*, pages 191–197. Springer, Cham, 2019.
48. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.
49. P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *stat*, 1050:20, 2017.
50. L. L. Wang, C. Bhagavatula, M. Neumann, K. Lo, C. Wilhelm, and W. Ammar. Ontology alignment in the biomedical domain using entity definitions and context. *arXiv preprint arXiv:1806.07976*, 2018.
51. X. Wang, H. Pham, P. Arthur, and G. Neubig. Multilingual neural machine translation with soft decoupled encoding. *arXiv preprint arXiv:1902.03499*, 2019.
52. Z. Wang and I. F. Cruz. AgreementMakerDeep results for OAEI 2021. In *Proceedings of the 16th Workshop on Ontology Matching*, pages 124–130, 2021.
53. J. Wu, J. Lv, H. Guo, and S. Ma. Daeom: A deep attentional embedding approach for biomedical ontology matching. *Applied Sciences*, 10(21):7909, 2020.
54. L. Wu, A. Fisch, S. Chopra, K. Adams, A. Bordes, and J. Weston. Starspace: Embed all the things! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.
55. Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.
56. C. Xiang, T. Jiang, B. Chang, and Z. Sui. Ersom: A structural ontology matching approach using automatically learned entity representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 2419–2429, 2015.
57. Y. Yu, X. Si, C. Hu, and J. Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.
58. Y. Zhang, X. Wang, S. Lai, S. He, K. Liu, J. Zhao, and X. Lv. Ontology matching with word embeddings. In *Chinese computational linguistics and natural language processing based on naturally annotated big data*, pages 34–45. Springer, 2014.
59. T. Zhao, X. Zhang, and S. Wang. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In L. Lewin-Eytan, D. Carmel, E. Yom-Tov, E. Agichtein, and E. Gabrilovich, editors, *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*, pages 833–841. ACM, 2021.