DRAL-OA: Results for OAEI 2025

Adishesh Gonibeed Ravishankar[1], Mehrnoosh Zaefi[1] and Srividya Bansal[1]

[1]*School of Computing and Augmented Intelligence, Arizona State University, Tempe, AZ, USA*

Abstract

This paper introduces the results of Deep Reinforcement Adaptive Learning for Ontology Alignment (DRAL-OA), a framework that uniquely applies a semi-supervised, deep reinforcement learning approach to efficiently align ontologies. By integrating both semantic and structural information, this work presents the results of our tool, DRAL-OA, on the challenging Anatomy dataset. The findings validate our framework, demonstrating that the reinforcement learning agent successfully optimizes concept embeddings.

Keywords

Ontology alignment, Ontology Matching, Deep Reinforcement Learning, Neural Network, Adaptive Learning

1. Presentation of the system

DRAL-OA is an ontology alignment system that uses a semisupervised machine learning technique that can predict similar source and target ontology classes based on their ontological structural hierarchy, meta-information, and syntactic structure without any background of domain knowledge, in contrast to existing learning-based approaches. In the following sections, we present the methodology behind the system and the results of the system's participation in the OAEI 2025 initiative.

1.1. State, purpose, general statement

Ontology alignment, also called ontology matching, is the procedure of establishing semantic links between entities from different ontologies or knowledge graphs. This task is essential for resolving the semantic heterogeneity that arises when various communities independently create ontologies, leading to conceptual and structural differences. The core function of alignment is to identify mappings between these distinct ontologies to enhance data interoperability and resolve ambiguity. By creating these semantic correspondences, the process facilitates effective knowledge sharing and enables seamless communication between systems built on different ontological foundations, making it a critical component of data integration.

1.2. Specific techniques used

Our proposed model is a novel, lightweight, semi-supervised deep reinforcement learning model named Deep Reinforcement Adaptive Learning for Ontology Alignment (DRAL-OA). The DRAL-OA model incorporates two main components: the preprocessing of the source and target ontologies and the deployment of a reinforcement learning agent as shown in Figure 1. We provide a detailed description of these components in the following sections.

(i) ***Data Preparation:*** The data processing pipeline is built upon the foundational principles of ONTOCONNECT [1], a domain-agnostic ontology alignment method that utilizes graph embedding with negative sampling, as presented by Chakraborty et al. Our approach adapts and improvises upon this framework to meet the specific requirements of our proposed system. In this step, a Java API named OWL API [2] and HermiT Reasoner [3] are used to extract meta information about a class/concept, such as IRI, label, restriction, parent, child, equivalent, and disjoint classes of each

OM 2025: The 20th International Workshop on Ontology Matching collocated with the 24th International Semantic Web Conference (ISWC 2025), November 2nd, 2025, Nara, Japan.

Editors: Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Cássia Trojahn, Sven Hertling, Huanyu Li, Pavel Shvaiko, Jérôme Euzenat

adishesh@asu.edu (A. G. Ravishankar); mzaefi@asu.edu (M. Zaefi); skbansa2@asu.edu (S. Bansal)

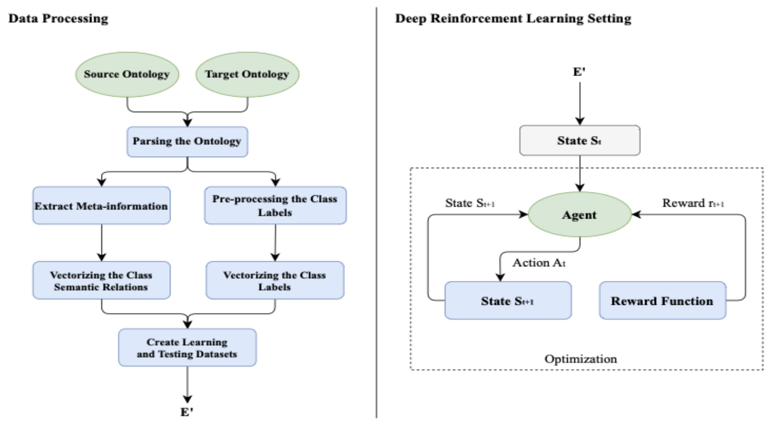

Figure 1: Proposed workflow overview

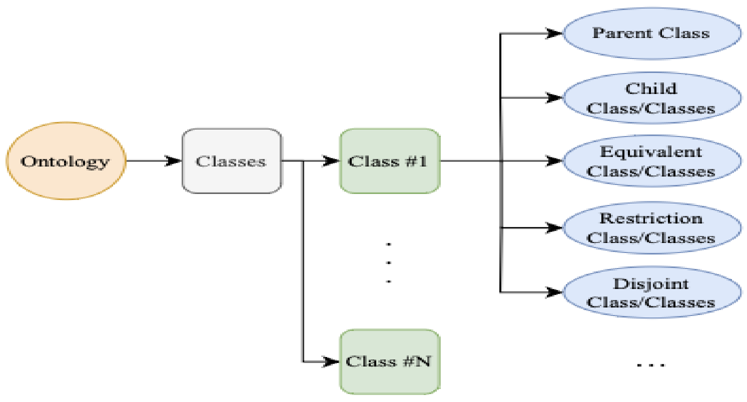

Figure 2: Structural information of a Source/Target Ontology

class/concept of the source ontology (S) and the target ontology (T).

(ii) ***Data Pre-processing:*** Several data pre-processing techniques are used on both source and target class/concept labels. Special characters and common stop-words in English are removed from the class/concept labels. Apart from stopwords, we have used tokenization, lemmatization, conversion of roman letters to numeric, etc.

(iii) ***Vector Generation:*** In this step, a pre-trained embedding model called fastText [4] developed by Facebook's AI Research (FAIR) lab is used on the source and the target ontology class/concept to generate vectors. It treats each word as composed of character n-grams. So, the vector for a word is made up of the sum of these character n-grams. It helps create a meaningful vector even when the dictionary word is not present in the model. The default dimension of the generated vector is 300.

(iv) ***Model Prediction:*** The word similarity is calculated by the cosine similarity between the source and target class/concept vectors. Next, the meta-information of the target ontology class is fed to the trained ontology alignment model which predicts a vector similar to one of the source classes. We use the cosine similarity to measure the meta similarity as well. A combined similarity i.e., the average of the word similarity and meta similarity, is used for the final prediction of similar class/concept.

(v) ***Vectorization of Classes Structural Relation*** After extracting the structural relations between the classes of each ontology, a graph embedding based on the input ontologies was developed. The graph captures various types of structural connections, including parent-child hierarchies, restrictions, disjoint classes, and equivalences, effectively illustrating the intricate relationships among the nodes. Next, we train the graph embedding and create a vector representation for the relation of each class using the node2vec model [5]. Node2vec is an algorithmic framework designed for representation learning on graphs, generating continuous feature representations for the nodes in any given graph.

(vi) ***Creation of Training and Testing Datasets*** Finally, to create the training and testing datasets, we used a semi-supervised approach, combining 50 percent data from an external resource with 50 percent data generated by our tuned graph embedding model.

1.3. Deep Reinforcement Adaptive Learning

Building on the innovative use of Deep Reinforcement Adaptive Learning (DRAL) in complex data environments [6], [7], we adapt and extend this method to the nuanced task of semantic ontology alignment. This advanced adaptation involves the strategic alignment of classes between a source ontology S and a target ontology T, using DRAL to intelligently navigate and optimize these mappings. Our learning approach, showed in Algorithm 1, diverges from traditional adversarial learning methods, leveraging a tailored DRAL algorithm that focuses on semantic precision and structural integrity rather than mere classification accuracy.

Environment: The DRAL environment is uniquely configured around the ontological structures, incorporating the semantic and the vector representations of ontological concepts derived from sophisticated embedding models. The agent acts on the current embedding E'_t, resulting in a new state E'_{t+1}. The effectiveness of each action is assessed by computing a reward based on how well the modified embedding E'_{t+1} facilitates improved alignment between ontological classes, enhancing the precision of ontology matching.

State: In our DRAL framework, the state is defined by the vector representation E' of a class from the source ontology. This representation encapsulates the current challenge faced by the RL agent, offering a detailed snapshot of the semantic attributes and potential alignment with the target ontology.

Action: Actions are defined as selecting one of the values in the source ontology representation E' and modifying it by either adding or subtracting a small value σ. The total number of actions is $|E'| \times 2$.

Reward Function: We have selected contrastive loss as the primary loss function for our project due to its ability to finely differentiate vector similarities, essential in ontology alignment. Formally, the reward r_{t+1} at a new state s_{t+1} is calculated as follows:

$$L_{i,j} = -\log\left(\frac{\exp(\text{sim}(z_i, z_j)/T)}{\sum_{k=1\ [k\neq t]}^{2N} \exp(\text{sim}(z_i, z_k)/T)}\right) \tag{1}$$

where $\text{sim}(z_i, z_j)$ represents the cosine similarity measure between the embeddings z_i and z_j, and T denotes the temperature parameter that scales the similarity scores. The function aims to increase the similarity of positive pairs while decreasing the similarity of negative pairs relative to an anchor, enhancing discrimination in the embedding space.

This method adapts the softmax function by adding a vector similarity measure through cosine distance and a temperature normalization factor to tune sensitivity to vector variations. Similar to CrossEntropyLoss but with a key distinction, contrastive loss computes the denominator values as the cosine distances from a positive example to negative samples, enhancing the model's discriminatory power. The goal is for similar vectors to approach a similarity of 1, thereby minimizing the loss to zero, while dissimilar pairs aim for a similarity near zero to maximize their contribution to the loss. This approach ensures that our model sharply distinguishes between matching and non-matching ontological terms, significantly improving alignment accuracy and effectiveness.

Algorithm 1 The Learning Process of DRAL-OA

```
Require: Ontologies representations $E' \in D$, parameters $\alpha$, $\beta$, and $\lambda$, learning rate $lr$, and terminal time $T$.
 1: Initialize state $s_t$ and memory $M$.
 2: while training is not finished do
 3:     $s_t \leftarrow E'$
 4:     for $t \in \{0, 1, ..., T\}$ do
 5:         Choose action $a_t$ according to current distribution $\pi(s_t)$
 6:         Perform $a_t$ on $s_t$ and get $(s_{t+1}, r_{t+1})$
 7:         $M \leftarrow M + (s_t, a_t, r_{t+1}, s_{t+1})$
 8:         $s_t \leftarrow s_{t+1}$
 9:         for each timestep $t$, reward $r$ in $M_t$ do
10:             $G_t \leftarrow \sum_{i=1}^{t} \lambda^i r_{i+1}$
11:         end for
12:         Calculate policy loss according to Equation 2
13:         Update the agent's policy according to Equation 3
14:     end for
15: end while
```

1.4. Optimization Algorithm

The primary aim of this research is to develop and refine an optimal action-selection strategy $\pi(s_t, a_t)$ tailored for the domain of ontology mapping. In this optimization, each timestep t necessitates the DRAL agent to modify the representation $s_t = E'_t$ of a class from the source ontology. The objective for the agent is to maximize the reward r_{t+1}, which is based on the updated representation $s_{t+1} = E'_{t+1}$ and the action a_t executed.

For the training and optimization of our agent, the REINFORCE algorithm is employed, utilizing a policy gradient approach to refine the mapping strategy. The agent's policy, parameterized by θ, is denoted as $\pi_\theta(s_t, a_t)$ and is evaluated using the loss function:

$$L(\theta) = \log(\pi_\theta(s_t, a_t) \cdot G_t) \tag{2}$$

Here, G_t represents the cumulative sum of discounted rewards, calculated as $\sum_{i=t}^{\infty} \lambda^{i-t} r_{i+1}$, where λ serves as the discount factor. This emphasizes the valuation of more immediate rewards, which are pivotal in our context of dynamic ontology mapping.

The updating of policy parameters θ is facilitated through gradient ascent, computed as follows:

$$\nabla\theta = lr\nabla\theta L(\theta) \tag{3}$$

In this expression, lr signifies the learning rate, a critical factor in managing the pace and effectiveness of policy updates. This structured optimization framework enables the RL agent to incrementally improve its decision-making processes, thereby enhancing its ability to propose mappings that are both semantically coherent and structurally sound. Ultimately, this approach aims to not only optimize immediate rewards but also to sustain the robustness and precision of mappings across heterogeneous ontological structures.

1.5. Adaptations made for the evaluation

The DRAL-OA pipeline was implemented using Python and subsequently containerized with Docker via the MELT platform to ensure ease of use and streamlined testing. We utilized the MELT Python web matcher, which exposes an HTTP endpoint, enabling the matching system to receive source and target RDF files and parameters through a URL-encoded form or multipart upload, directly returning the final alignment. Although the system generally operated effectively, we encountered intermittent

Table 1
DRAL-OA performance in the Anatomy track

Matcher	Runtime	Size	Precision	F-Measure	Recall	Recall+	Coherent
DRAL-OA	877	1509	0.83	0.828	0.827	0.56	-

output errors and compatibility issues when running the container on different operating systems. To mitigate these inconsistencies and guarantee a clean output, an alternative deployment was provided as a Google Colab notebook. This approach, while incurring a marginal increase in execution time, ensured a stable and reproducible environment.

1.6. Link to the system and parameters file

The DRAL-OA code is available on GitHub: https://github.com/AGR19/DRAL-OA and on Google Colab : https://colab.research.google.com/drive/1ApvzO1V0dZGjkARnRuYS_srvSP2-zwCt?usp=sharing

1.7. Link to the set of provided alignments

The DRAL-OA result is published on http://oaei.ontologymatching.org/2025/results/anatomy/index.html . The result is also available on GitHub: https://github.com/AGR19/DRAL-OA

2. Results

We have tested DRAL-OA on the Anatomy [8] data set published by OAEI. Three inputs provided in the OAEI System: source ontology, target ontology, and reference file containing correct matches to calculate precision , recall and F-Measure . The DRAL-OA system yields satisfactory results with a precision of 83%, recall of 82.7%, and F-measure of 82.8%. Table 1 gives a summary of the result of DRAL-OA on the Anatomy data set.

3. General comments

In the ontology alignment task, the system generates multiple candidate matches for each class from the source ontology, ranked by similarity scores. The evaluation considers the top-k highest-scoring predictions. During our internal testing, we adopted a top-5 (k=5) evaluation, a more lenient approach where a match is considered correct if the true correspondence appears anywhere within the five highest-ranked predictions. This method is useful for maximizing recall, ensuring the correct answer is included in the candidate set, though potentially with lower precision. In contrast, the official evaluators calculated the final results using a top-1 (k=1) setting. This is the most restrictive metric, where only the single best match is considered, and a prediction is deemed correct only if it is ranked first, thereby prioritizing the highest precision at the expense of recall.

4. Conclusion

The paper presents DRAL-OA, a novel system for ontology alignment that uses a semi-supervised, deep reinforcement learning (DRL) approach. The system effectively integrates both semantic information (from class labels) and structural information (from relationships like parent-child, restrictions, etc.) to map concepts between different ontologies.

The core of the DRAL-OA framework is a reinforcement learning agent trained to intelligently optimize the vector representations (embeddings) of concepts. By modifying these embeddings based on a reward signal derived from a contrastive loss function, the agent improves the quality of the alignments.

When evaluated on the challenging Anatomy dataset from the Ontology Alignment Evaluation Initiative (OAEI), the DRAL-OA system demonstrated strong performance, achieving a precision of 83 percent, a recall of 82.7 percent, and an F-measure of 82.8 percent. These results validate the effectiveness of applying deep reinforcement learning to the complex task of ontology matching.

Declaration on Generative AI

During the preparation of this work, the authors used Gemini in order to check and improve grammar and spelling of initial drafts. Afterwards, the authors reviewed and edited the content and take full responsibility for the publication's content.

References

[1] J. Chakraborty, S. Bansal, B. Yaman, L. Virgili, K. Konar, Ontoconnect: Unsupervised ontology alignment with recursive neural network, in: Proceedings of the 36th ACM/SIGAPP Symposium on Applied Computing, SAC 2021, Gwangjiu, South Korea, March 22-26, 2021, 2021. (In Press).

[2] M. Horridge, S. Bechhofer, The owl api: A java api for owl ontologies, Semantic Web 2 (2011) 11–21.

[3] B. Motik, R. Shearer, I. Horrocks, Optimized reasoning in description logics using hypertableaux, in: International Conference on Automated Deduction, Springer, 2007, pp. 67–83.

[4] P. Bojanowski, E. Grave, A. Joulin, T. Mikolov, Enriching word vectors with subword information, Transactions of the Association for Computational Linguistics 5 (2017) 135–146.

[5] J. Leskovec, A. Grover, node2vec: Scalable feature learning for networks, https://snap.stanford.edu/node2vec/, 2016.

[6] A. Mosallanezhad, G. Beigi, H. Liu, Deep reinforcement learning-based text anonymization against private-attribute inference, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 2360–2369.

[7] A. Mosallanezhad, M. Karami, K. Shu, M. V. Mancenido, H. Liu, Domain adaptive fake news detection via reinforcement learning, in: Proceedings of the ACM Web Conference 2022, 2022, pp. 3632–3640.

[8] OAEI, Oaei 2025 anatomy track, http://oaei.ontologymatching.org/2025/anatomy/index.html, 2025.