JSimELHExplainer: A Robust JAVA Library for Explainable Semantic Similarity for ELH Description Logic Ontology

Teeradaj Racharak[1,*], Watanee Jearanaiwongkul[1]

[1]Advanced Institute of So-Go-Chi (Convergence Knowledge) Informatics, Tohoku University, Miyagi, Japan

Abstract

We present a newly developed Java library that implements a neuro-symbolic framework for computing semantic similarity between concepts in Description Logic $\mathscr{ELH}$ ontologies. This library provides an implementation of a hybrid approach combining a structural-based method in ontology reasoning with distributional semantics derived from pre-trained word embeddings. It supports efficient similarity computation and interpretable explanations for the results. Designed with scalability in mind, it guarantees polynomial-time execution and supports large-scale ontologies with thousands of concepts and complex hierarchical structures. For explainability, it produces fine-grained explanations by identifying the contributing primitive and existential concept pairs, as well as the semantic alignments found in the embedding space. These explanations help users understand why a similarity score is given, making the results transparent and auditable. The API is embedding-agnostic and compatible with a wide range of vector space models, including static embeddings (e.g., Word2Vec) and contextualized models (e.g., BERT). This tool enables the development of explainable, knowledge-driven AI systems in domains where both structured ontological modeling and contextual semantic understanding are essential.

Keywords

Semantic Similarity, Non-standard Reasoner, Ontology, Explainable AI, JAVA API

1. Introduction

Measuring semantic similarity between ontology concepts is essential in applications such as clinical decision support, semantic search, and ontology alignment [1, 2]. Structural-based approaches in ontology reasoning offer interpretability and logical consistency but are limited in capturing hidden or implicit relationships. In contrast, embedding-based methods leverage distributional semantics from large text corpora to model contextual meaning, enabling generalization, robustness to lexical variation, and the ability to capture similarity even in sparsely modeled ontologies.

To bridge the strengths of both approaches, we present a Java API implementing a neuro-symbolic framework (originally proposed in [3]) for concept similarity in Description Logic $\mathscr{ELH}$. The API combines structural reasoning over ontology hierarchies with pre-trained word embeddings to compute similarity scores that are both explainable and semantically enriched. It also supports automatic generation of human-readable explanations, promoting transparent and trustworthy AI.

ISWC 2025 Companion Volume, November 2–6, 2025, Nara, Japan

*Corresponding author.

racharak@tohoku.ac.jp (T. Racharak); watanee@tohoku.ac.jp (W. Jearanaiwongkul)

0000-0002-8823-2361 (T. Racharak); 0000-0003-2101-1625 (W. Jearanaiwongkul)

1.1. Algorithmic Design and Implementation

Our library, called *JSIM$\mathcal{ELH}$Explainer*, implements an efficient and explainable concept similarity measure within the $\mathcal{ELH}$ fragment of Description Logic. The core algorithm is designed in accordance with the neuro-symbolic framework proposed in the prior theoretical works [3, 4], with a practical focus on scalability, modularity, and extensibility.

1.1.1. Supported Ontology Type

The current implementation supports $\mathcal{ELH}$ ontologies with unfoldable TBoxes. This ensures that no cyclic definitions exist, making the ontology amenable to tree-based structural analysis.

1.1.2. Embedding Model Independence

The similarity computation module is designed to be agnostic to the choice of embedding models. Any word embedding model—static or contextualized, domain-specific or general-purpose—can be used as long as it provides a mapping from concept labels (strings) to vectors in a real-valued vector space.

1.1.3. Algorithm and API's Development

Definition 1. *[Homomorphism Degree subject to Embedding [3]] Let* $\mathsf{CN}^{\mathsf{pri}}$ *and* RN *be a set of primitive concept names and role names in an ontology, respectively. Also, let* $\mathbf{T}^{\mathcal{ELH}}$ *be a set of all* $\mathcal{ELH}$ *description trees,* $\mathcal{E}$ *be a set of (possibly empty) pre-trained embeddings, and* $\mathscr{M} : \mathsf{CN}^{\mathsf{pri}} \cup \mathsf{RN} \to \mathcal{E}$ *be a mapping. The* homomorphism degree subject to $\mathcal{E}$ *(denoted by* hd_ϵ*) is a function* $\mathsf{hd}_\epsilon : \mathbf{T}^{\mathcal{ELH}} \times \mathbf{T}^{\mathcal{ELH}} \to [0,1]$ *defined inductively as follows:*

$$\mathsf{hd}_\epsilon(\mathcal{T}_C, \mathcal{T}_D) \coloneqq \mu \times \mathsf{p\text{-}hd}_\epsilon(\mathscr{P}_C, \mathscr{P}_D) + (1-\mu) \times \mathsf{e\text{-}set\text{-}hd}_\epsilon(\mathscr{E}_C, \mathscr{E}_D) \tag{1}$$

where $\mu \coloneqq \frac{|\mathscr{P}_C|}{|\mathscr{P}_C \cup \mathscr{E}_C|}$*, also,* $\mathscr{P}_C$*,* $\mathscr{E}_C$ *denotes a set of primitive concepts and a set of existentials, respectively, on (unfolded) description tree* $\mathcal{T}_C$*;*

$$\mathsf{p\text{-}hd}_\epsilon(\mathscr{P}_C, \mathscr{P}_D) \coloneqq \begin{cases} 1 & \text{if } \mathscr{P}_C = \emptyset \\ 0 & \text{if } \mathscr{P}_C \neq \emptyset \\ & \text{and } \mathscr{P}_D = \emptyset \\ \dfrac{\sum_{A \in \mathscr{P}_C} \max_{B \in \mathscr{P}_D} \{\mathfrak{s}_\epsilon(A,B)\}}{|\mathscr{P}_C|} & \text{otherwise,} \end{cases} \tag{2}$$

$$\mathfrak{s}_\epsilon(A,B) \coloneqq \begin{cases} 1 & \text{if } A = B \\ \max\{\cos(\mathscr{M}(A), \mathscr{M}(B)), 0\} & \text{otherwise,} \end{cases} \tag{3}$$

where $\cos(\cdot,\cdot)$ *represents the cosine similarity,*

$$\mathsf{e\text{-}set\text{-}hd}_\epsilon(\mathscr{E}_C, \mathscr{E}_D) \coloneqq \begin{cases} 1 & \text{if } \mathscr{E}_C = \emptyset \\ 0 & \text{if } \mathscr{E}_C \neq \emptyset \\ & \text{and } \mathscr{E}_D = \emptyset \\ \dfrac{\sum_{\exists r.X \in \mathscr{E}_C} \max_{\epsilon_j \in \mathscr{E}_D} \{\mathsf{e\text{-}hd}_\epsilon(\exists r.X, \epsilon_j)\}}{|\mathscr{E}_C|} & \text{otherwise,} \end{cases} \tag{4}$$

where ϵ_j is an existential; and

$$\text{e-hd}_\epsilon(\exists r.X, \exists s.Y) := \gamma_\epsilon(r,s) \times (v + (1-v) \times \text{hd}_\epsilon(\mathcal{T}_X, \mathcal{T}_Y)) \tag{5}$$

where $0 \leq v < 1$; and

$$\gamma_\epsilon(r,s) := \begin{cases} 1 & \text{if } r = s \\ \frac{\sum_{r' \in \mathcal{R}_r} \max_{s' \in \mathcal{R}_s} \{\cos(\mathcal{M}(r'), \mathcal{M}(s')), 0\}}{|\mathcal{R}_r|} & \text{otherwise,} \end{cases} \tag{6}$$

where $\mathcal{R}_r$, $\mathcal{R}_s$ denotes a set of all r's super roles and a set of all s's super roles, respectively. Note that a set of all r's super roles, denoted by $\mathcal{R}_r$, is defined as $\mathcal{R}_r = \{s \in \mathsf{RN} \mid r \sqsubseteq^* s\}$ and, $r \sqsubseteq^* s$ if $r = s$ or $r_i \sqsubseteq r_{i+1} \in \mathcal{T}$ where $1 \leq i \leq n, r_1 = r, r_n = s$, and $*$ is a transitive closure.

From the above definition, we describe the main procedure of our similarity procedure implemented in our library here. Given a target ontology and two concepts whose similarity is to be computed, the similarity procedure in our library proceeds with respect to the following main steps:

1. **Unfolding the TBox:** The ontology is preprocessed by replacing defined concept names with their definitions recursively until all concept expressions are written in terms of only primitive concepts (i.e., concept names that are not defined in the TBox).
2. **Description Tree Construction:** Each unfolded concept is translated into a description tree, a syntactic representation that reflects the hierarchical and conjunctive structure of the concept expression based on $\mathscr{ELH}$ constructors.
3. **Structural Comparison:** A syntactic comparison is performed between the two description trees as described in Definition 1. The algorithm recursively computes a similarity degree by combining structural correspondence with lexical proximity (i.e. the used embeddings).
4. **Embedding-based Similarity:** For each concept or role label encountered, the algorithm uses the chosen embedding model (cf. Mapping $\mathscr{M}$ in Definition 1) to compute a cosine similarity between matching nodes or edges. This allows for approximate matching even when labels differ lexically but are semantically related.
5. **Similarity Aggregation:** The final score is obtained by aggregating the local similarity scores of the matched substructures, weighted appropriately based on tree depth and logical operators.

The algorithm has been mathematically proven to run in polynomial time with respect to the size of the ontology and the input concept descriptions. Termination is guaranteed due to the finite nature of the unfolding and tree construction processes, as well as the bounded number of structural comparisons during similarity computation.

2. Core API Design

The core APIs are designed to facilitate our concept similarity computation, explanation generation, and easy integration with OWL ontologies and external embedding models. The library is available online at: https://github.com/realearn-people/sim-elh-explainer-jar

Below is an overview of the key functions provided in our development:

1. **Instantiation of JSIM$\mathscr{ELH}$ Explainer:** The first step is to instantiate a SimExplainer object for loading the input ontology into a targeted Java project. Following [3, 4], all primitive concept names and role names could be also pre-configured in a separate setting, called *Preference Profile* [4]. In addition, our API supports two kinds of ontology file extensions: .owl and .krss.
2. **Concept Name Retrieval:** This function allows to retrieve all concept names from the ontology.
3. **Similarity Measure:** This function measures similarity between two concepts.
4. **Description Tree Retrieval:** This function returns a description tree of the given concept. This can be represented in JSON for machine readability and in ASCII for human readability.
5. **Explanation Retrieval:** This function returns a 4-tuple representing the explanation of detected similarity consisting of (1) the *homomorphism degree* from a concept to another concept, (2) a *list of primitive concept pairs* that contribute to the similarity score, with each pair consisting of one concept from the first input and one from the second, (3) a *list of existential concept pairs* contributing to the similarity score, where each pair consists of one existential concept from each of the two input concepts, and (4) an applied *embedding* map in which each key is a pair of existential or primitive concepts—one from each input concept—that contributes to the overall similarity score. For the 4th one, the corresponding value is a set of pairs, where each pair represents a similarity found within the embedding space between roles or primitive concepts.

We provide the output of explanation retrieval in both JSON and ASCII for flexibility of use of the library. Note that the final similarity score is derived by averaging the homomorphism degrees (Definition 1) from both the forward and backward directions of the respected description trees.

#	Insight Comparison	Concept Pairs	Similarity Score
1	2 concepts under the same class	Son, SonInLaw*	0.97
		Son, Uncle	0.897
2	2 concepts under different classes but same level	SonInLaw, DaughterInLaw*	0.8
		SonInLaw, Daughter	0.68
3	2 concepts that are direct class-subclass	Woman, Aunt	0.9
4	2 concepts that are not direct class-subclass	Female, GrandFather	0.53

Table 1
Insight analysis on the experiments based on the family ontology, where * denotes that they yield higher scores due to more numbers of shared features.

3. Experiments and Conclusions

We evaluated our similarity framework using SNOMED CT and the family ontology [5]. We aimed to validate that concepts sharing more features should yield a higher similarity degree. To manage complexity of SNOMED CT, we focused on two top-level categories: Clinical Finding and Procedure. We randomly sampled 206 concepts (0.5%) from each category, forming three test sets: $C_1 \times C_1$, $C_2 \times C_2$, and $C_1 \times C_2$, where C_1 and C_2 represent the Clinical Finding and Procedure samples, respectively. *Our findings confirmed that concept pairs from the same category consistently received higher similarity scores than cross-category pairs, in line with expectations based on ontological structure and subsumption.* Table 1 shows some insights for the chosen pairs in the family ontology. In the future, we plan to validate on other aspects such as usability. This could be done by involving users or developers to gather feedback.

Declaration of Generative AI and AI-assisted Technologies

The authors did not use any generative AI and AI-assisted technologies for writing the article. The authors thus take full responsibility for the content of the publication.

References

[1] F. Z. Smaili, X. Gao, R. Hoehndorf, Opa2vec: combining formal and informal content of biomedical ontologies to improve similarity-based prediction, Bioinformatics 35 (2019) 2133–2140.

[2] L. Zhao, J. Wang, L. Cheng, C. Wang, Ontosem: an ontology semantic representation methodology for biomedical domain, in: 2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), IEEE, 2020, pp. 523–527.

[3] T. Racharak, On approximation of concept similarity measure in description logic ELH with pre-trained word embedding, IEEE Access 9 (2021) 61429–61443. URL: https://doi.org/10.1109/ACCESS.2021.3073730. doi:10.1109/ACCESS.2021.3073730.

[4] T. Racharak, B. Suntisrivaraporn, S. Tojo, Personalizing a concept similarity measure in the description logic ELH with preference profile, Comput. Informatics 37 (2018) 581–613. URL: https://doi.org/10.4149/cai_2018_3_581. doi:10.4149/CAI_2018_3_581.

[5] R. Stevens, M. Stevens, A family history knowledge base using owl 2., in: Owled, volume 432, 2008.