

mOWL: Python library for machine learning with biomedical ontologies

Fernando Zhapa-Camacho

<2023-02-13 Mon>

Ontologies

Ontologies contain information on different axes:

- Classes and relations
- Domain vocabulary
- Descriptions
- Axioms

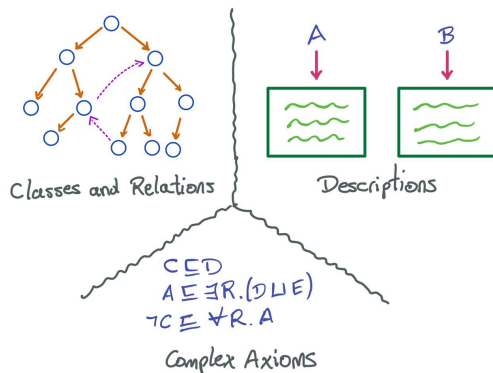


Figure: Components of ontologies

Ontologies use

- Ontologies provide background knowledge for a domain

Ontologies use

- Ontologies provide background knowledge for a domain
- That knowledge can be leveraged by a machine learning model

mOWL: Ontology-centric-designed library

- Python library to integrate ontologies and machine learning models

mOWL: Ontology-centric-designed library

- Python library to integrate ontologies and machine learning models
- Interfaces OWLAPI (which is written in Java)

mOWL: Ontology-centric-designed library

- Python library to integrate ontologies and machine learning models
- Interfaces OWLAPI (which is written in Java)
- Ontology Management

mOWL: Ontology-centric-designed library

- Python library to integrate ontologies and machine learning models
- Interfaces OWLAPI (which is written in Java)
- Ontology Management
- Ontology Transformation

mOWL: Ontology-centric-designed library

- Python library to integrate ontologies and machine learning models
- Interfaces OWLAPI (which is written in Java)
- Ontology Management
- Ontology Transformation
- Embedding Generation

mOWL: Ontology-centric-designed library

- Python library to integrate ontologies and machine learning models
- Interfaces OWLAPI (which is written in Java)
- Ontology Management
- Ontology Transformation
- Embedding Generation
- Embedding Post-processing

mOWL provides methods to manipulate ontologies:

- creation
- modification
- reasoning

mOWL interfaces the OWLAPI

Ontology Transformation

mOWL provides functionalities to transform ontologies and/or extract information in different ways:

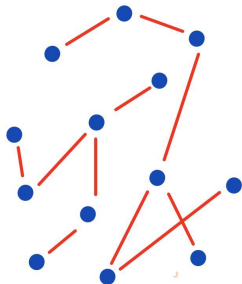


Figure: To graphs

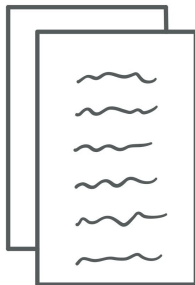


Figure: To text

Axioms

$C \sqsubseteq D$
 $C \sqsubseteq \exists R.E$
 $\exists R.F \sqsubseteq G$

Figure: To axioms

Ontology Transformation: Graphs

- We call *projection* to the process of transforming an ontology into a graph.
- There are many methods to project an ontology.
- In general, every projection undergoes some kind of *loss of information*

Graph projections in mOWL

mOWL provides several projection methods:

- Taxonomy
- Taxonomy + existential relations
- DL2Vec
- OWL2Vec*

Ontology Transformation: Graphs

mOWL code

```
from mowl.projection import DL2VecProjector  
projector = DL2VecProjector(True)
```

Ontology Transformation: Text

This approach uses the syntactic information of the axioms and generates text sentences out of them.

- Axioms are defined over a syntax (symbols, operators, ...)
- Syntactic elements can be represented as words
- Onto2Vec, OPA2Vec

Ontology Transformation: Axioms

Some methods would require preprocessing of axioms:

- Normalization (ELEmbeddings)
- Grouping into common structural patterns (FALCON)