mOWL: Python library for machine learning with biomedical ontologies

Fernando Zhapa-Camacho

<2023-02-05 Sun>

Learning Objectives

- Review about different components of ontologies
- Understand the use of different components of ontologies in a machine learning setting
- Learn to use mOWL in order to work with ontologies in machine learning.

Ontologies

Ontologies contain information on different axes:

- Classes and relations
- Domain vocabulary
- Descriptions
- Axioms

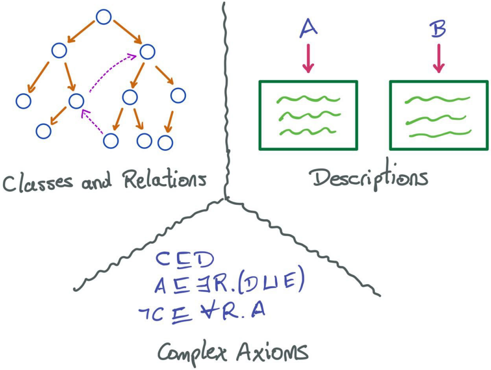

Figure: Components of ontologies

Ontologies

Ontology axioms

Ontology axioms are defined over a signature $\Sigma = (\mathrm{C}, \mathrm{R}, \mathrm{I})$ where C is a set of concept names, R is a set of role names and I is a set of individual names.

Concepts:

- Person
- Mother
- Child

Roles:

- hasChild
- hasSibling
- hasFriend

Individuals:

- Mary
- John
- Bob

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
  - Embedding generation
  - $f : \Sigma \to \mathbb{R}^n$

Ontologies use

- Ontologies provide background knowledge for a domain
- That knowledge can be leveraged by a machine learning model
  - Embedding generation
  - $f : \Sigma \rightarrow \mathbb{R}^n$
- In recent years, many machine learning models have been developed to *embed* ontologies.

What does *embed* an ontology mean?

- An embedding is a structure-preserving mapping between a source and a target structures.

What does *embed* an ontology mean?

- An embedding is a structure-preserving mapping between a source and a target structures.
- Usually, the target structure is more suitable to perform operations on the entities.

What does *embed* an ontology mean?

- An embedding is a structure-preserving mapping between a source and a target structures.
- Usually, the target structure is more suitable to perform operations on the entities.
- Embedding ontologies into $\mathbb{R}^n$ might be useful to perform operations such as:
  - Similarity computation
  - Inferences

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

Ontology Management

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:

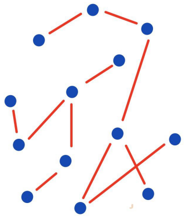

Figure: To graphs

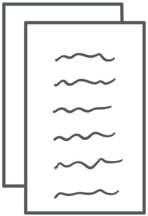

Figure: To text

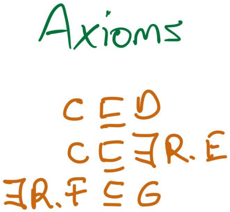

Figure: To axioms

Ontology Transformation

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

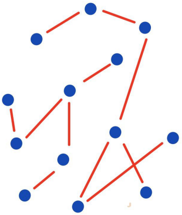

Figure: To graphs

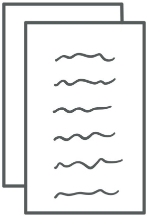

Figure: To text

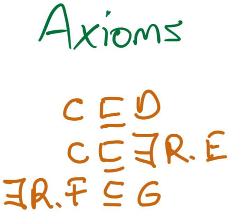

Figure: To axioms

- Each transformation result can be used as input of a machine learning model.

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(bidirectional_taxonomy=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, ELBoxEmbeddings)
- Grouping into common structural patterns (FALCON)

mOWL

- Accessible from:
  - GitHub: `https://github.com/bio-ontology-research-group/mowl/tree/main/mowl`
  - PyPi: `pip install mowl-borg`