Babylon - KE Coding Test

Enrico Bignotti

27th February 2019

1 My approach

My approach relied mainly on exploiting the data structure of Python DataFrames, since, as described in Section 2, I could not manage to convert the .tsv files in RDF/OWL models.

The first step was to considering how similar were the ontologies. The simplest way was to take advantage of the methods for checking whether two DataFrames are equal. Before doing that, I ordered them according to their predicates, so that I could recreate a series of OWL classes with one predicate per row, where the first predicate is `rdf:type`, and the others follow accordingly. Thus, I could return a third DataFrame containing only those classes which where not shared (thankfully just 4: 'JournalOrPeriodic', 'Journal', 'Conference', and 'Congress').

All these classes where equivalent in pairs, as they had the same label and comment predicate, so the mapping task was essentially just adding a `owl:equivalenceClass` between each pair of these classes and then add them to either ontologies. It could have been easy enough to do it manually, but mapping them like that would have definitely not scaled in a real environment.

So, I designed a script that checked the predicates, i.e., the rows, and took advantage of their order within the classes, namely that labels followed comments, so if a comment was the same with respect to any other comment in the DataFrame, then the labels had to be checked. If they were also the same, then, since there were no other predicates save for `rdf:type`, the classes had to be equal.

All these equality predicates could then be saved in a mapping file and added to either the ontologies to create the desired ontology ammping the initial two.

2 Problems

My main issue throughout the exercise was that I could not find a consistent way to adapt the .tsv files to RDF—OWL and thus produce an actual ontology, resorting to developing a manual version on my own, which in the end did not really work.

A second issue was that I could not exploit and find in time an authoritative source to use for the mapping in helping choosing whether to consider the equivalence more of a synonymy or an actual equivalence. For instance, I considered using WordNet[1] and the NLTK[2] module to check whether the classes belonged to the same synset, and where thus synoyms. Unfortunately, the classes were only sister terms, which means that they were coordinate terms, i.e., they shared the same hypernym; as such, they were neither equivalent nor synonyms. Overall, these problems forced me to use the simple equivalence measure described in Section 1.

3 Improvements

The main improvement would be to be actually able to produce an ontology and do a more semantic mapping, and not relying solely on exploiting syntactic similarities between the ontologies.

An interesting source to exploit as a mapping resource would most likely be the Semantic Publishing and Referencing Ontologies (SPAR)[3]. SPAR is a suite of ontology modules for the creation of comprehensive machine-readable RDF metadata in the publishing domain. The different classes could be compared for alignment and their semantics enriched by reusing these ontologies structure.

Additionally, some NLP approaches such as word embedding could be used on descriptions and labels which might have been not exactly the same but closely related, thus capturing more nuances from the data.

[1] `https://wordnet.princeton.edu/`
[2] `http://www.nltk.org/howto/wordnet.html`
[3] `http://www.sparontologies.net/`