A Clustering Based Approach for Type Discovery in RDF Data Sources

Kenza Kellou-Menouer*, Zoubida Kedad*

*PRISM - University of Versailles Saint-Quentin-en-Yvelines,
45 avenue des Etats-Unis , Versailles, France
kenza.menouer@prism.uvsq.fr, zoubida.kedad@prism.uvsq.fr

Querying and exploiting RDF(S)/OWL data sources requires information about the resources and properties they contain. Without a description of the data set, it is difficult to target the relevant properties and resources, and browsing data sets in order to understand their content can be a tedious process. One important feature of RDF(S)/OWL data sources is that they are not organized according to any predefined schema. They are structureless by nature and the languages used to describe data on the Web do not impose any constraints or restrictions on the properties describing resources.

Our goal is to discover missing type definitions in a RDF(S)/OWL data set. We propose a clustering based approach where entities are grouped according to their similarity. The similarity between two given entities is evaluated considering their respective sets of properties using Jaccard similarity.

Our requirements for type discovery are the followings: firstly, the number of types is not known in advance, and secondly, the data sets are evolving, large and may contain noise. The most suitable grouping approach is density-based clustering, introduced by Ester et al. (1996), because it is robust to noise, deterministic and it finds classes of arbitrary shape, which is useful for data sets where resources are described with heterogeneous property sets. In addition, unlike the algorithms based on k-means and k-medoid, the number of classes is not required.

In order to speed up the clustering process, and especially to perform successive executions with different parameters values (the maximum radius of neighborhood ε and the minimum number of neighbors for an entity $MinPts$), we perform once and for all the calculation of the nearest neighbors of each entity. To this end, we index the data and we order the entities according to their similarity. We store a neighborhood matrix containing for each entity the ordered list of its neighbors, as well as the distance between this entity list and the number of the line representing the index of an entity. Thanks to the indexing of the data and the ordering of the neighbors according to the distance, it is not necessary to go through the entire matrix to find the neighbors of a given entity, and the complexity of neighbors search becomes linear o(n).

We have performed some experiments on existing data sets to assess the quality of the inferred types. We have extracted the existing type definitions from our data sets and considered them as a gold standard. Then we have run our algorithm on the data sets without the type definitions and evaluated for each of the inferred classes precision and recall. We have annotated each inferred class with the most frequent type definition of its entities. For each type label T_r