Semsearch 2010 Entity Search Track
Scoring Model for Entity Search on RDF Graphs

Daniel M. Herzig
Institute AIFB, Karlsruhe Institute of Technology
76128 Karlsruhe, Germany
herzig@kit.edu

Thanh Duc Tran
Institute AIFB, Karlsruhe Institute of Technology
76128 Karlsruhe, Germany
ducthanh.tran@kit.edu

ABSTRACT

We describe an approach for scoring entities of an RDF graph for a given keyword query. The implementation of this approach was used to solve the task of entity search. The resulting system participated at the Semsearch 2010 Entity Search Track[1].

1. SCORING ENTITIES

We regard the subject of an RDF triple as an entity, which is identified by its URI. The set of all triples having the same subject is seen as the description of this entity. We divide this set of triples into three disjoint sets of triples, namely into the *label set* L, the *datatype set* D, and the *object set*. The *label set* L comprises all triples having a predicate, which is a sub-property of *rdfs:label* or which has a local name containing "name" or "label". The local name was taken into account, because there is often no schema information available. The transitivity of the *rdfs:subPropertyOf* relation was considered. The *datatype set* D consists of all triples having a literal as the object and are not an element of the *label* set. The other triples, i.e. those having a resource as the object, are elements of the *object* set, which we do not regard any further.

In order to rank entities for a given keyword query q, we calculate a score for each entity e. This score, $score(e, q)$, is a weighted sum over the scores of the *label* set $s_L(e, q)$ and the score of the *datatype* set $s_D(e, q)$. For practical reasons, we add a *graph* score $s_G(q)$, in order to avoid zeros, which is just an aggregation of the relative term frequencies in the entire RDF graph G. The higher the score the better fits the entity e to the query q. We regard the keyword query as a set of terms, $q = (t_1, \ldots, t_n)$. The $score(e, q)$ is calculated as follows, where $\mathrm{tf}(t, x)$ denotes the term frequency of term t in x and $|y|$ denotes the number of terms in y.

$$score(e, q) = \lambda_L \cdot s_L(e, q) + \lambda_D \cdot s_D(e, q) + \lambda_G \cdot s_G(q)$$

$$s_L(e, q) = \sum_{l \in L} \frac{\left(\sum_{t \in q} \mathrm{tf}(t, l)\right)^2}{|q|} \cdot \sum_{t \in q} \frac{tf(t, l)}{|l|} \cdot \left(1 - \frac{tf(t, G)}{|G|}\right)$$

$$s_D(e, q) = \sum_{l \in D} \frac{\left(\sum_{t \in q} \mathrm{tf}(t, l)\right)^2}{|q|} \cdot \sum_{t \in q} \frac{tf(t, l)}{|l|} \cdot \left(1 - \frac{tf(t, G)^2}{|G|}\right)$$

The scoring models basically sum over all elements of a set. For each element, the model rewards the number of term matches quadratically compared to the length of the query. Further, it captures how good a term matches the considered literal or label and discounts the value depending on the number of occurrences in the entire graph G. Here is the only difference between the scores s_D and s_L, for s_L discounting is only linear.

We indexed the billion triple challenge 2009 dataset[2] using Lucene as the underlying inverted index. Each triple was indexed as one document. Since there was no training data available, we chose $\lambda_L = 0.8$, $\lambda_D = 0.15$, and $\lambda_C = 0.05$ as weights without further tuning.

2. CONCLUSION AND OUTLOOK

Although datatype properties and labels play a crucial role when searching for entities in an RDF graph, the current model neglects all other features of the RDF graph. Especially, the structure of the graph, i.e., the connections between entities, contains valuable information. Therefore, the goal of our future work is to extend the model and take these aspects of the RDF graph into account.

[1] `http://km.aifb.kit.edu/ws/semsearch10/#eva`

WWW2010, April 26-30, 2010, Raleigh, North Carolina.

[2] `http://vmlion25.deri.ie/`