A *Semantic* Browser for Linked Open Data

Alexander Seeliger, Heiko Paulheim

Technische Universität Darmstadt, Germany
{seeliger,paulheim}@ke.tu-darmstadt.de

Abstract. Although the Semantic Web was originally designed as a "web for machines", the growing wealth of information in Linked Open Data has become interesting for human users as well. Consequently, quite a few browsers for Linked Open Data have recently been developed. However, despite being developed for the semantic web, those browsers often present alphabetically ordered lists of facts, without respecting the semantics of the data.

In our submission to the Semantic Web Challenge, we present a *semantic* browser for the semantic web[1], which aims at presenting facts from Linked Open Data in semantically coherent groups. This paper introduces the main algorithms as well as an evaluation of the browser with end users.

Keywords: Linked Open Data, User Interface, Browser, Semantic Interface

1 Introduction

Although the Semantic Web was originally designed as a "web for machines", the growing wealth of information in Linked Open Data has become interesting for human users as well. Consequently, quite a few browsers for Linked Open Data have recently been developed. Furthermore, most servers for Semantic Web data, such as *D2R*[2] or *Virtuoso*[3], have basic browsing capabilities. The latter provides the front-end to the popular DBpedia dataset [1], which lists all triples alphabetically, based on the predicate's URI.

Disco[4] [3], *Tabulator*[5] and [2] are some of the most popular browsers. Disco and Tabulator, like many similar systems, essentially present lists of facts, without any logical grouping and/or filtering. Thus, there are very few systems that actually provide a *semantic* view on the data. One rare exception is *aemoo*[6] [9],

[1] Live demo: http://kebap.ke.informatik.tu-darmstadt.de:8080/semantic-browser/
[2] http://d2rq.org/d2r-server
[3] http://virtuoso.openlinksw.com/
[4] http://www4.wiwiss.fu-berlin.de/bizer/ng4j/disco/
[5] http://www.w3.org/2005/ajar/tab
[6] http://wit.istc.cnr.it/aemoo

which extracts a number of core facts about an entity and augments them with text snippets from Wikipedia.

The main idea of our semantic browser is to get the sense of specific facts. Facts that are semantically related should be grouped together. For example, for a country, there are facts about the politics (the parties, the prime minister, etc.), the geography (cities, states, etc.), the culture, the climate, and so on. Our approach aims at grouping those facts automatically so that users can get a better view of the data and find information more quickly.

2 Approach

Our approach foresees several steps for semantically grouping facts. First, the data is retrieved, then, the similarity of the different statements is determined. Based on that similarity, groups are formed and labeled.

2.1 Preprocessing Datasets

Data in Linked Open Data is stored as RDF data, i.e., triples of a subject, a predicate, and an object [7]. When retrieving data about a resource from a Linked Open Data endpoint, in most cases, two types of triples are retrieved:

- Triples that have the retrieved resource as their subject. For example, for the resource `Darmstadt`, such a triple is `:Darmstadt :country :Germany`.
- Triples that have the retrieved resource as their object. For `Darmstadt`, such a triple would be `:TU_Darmstadt :city :Darmstadt`.

We aim at presenting both types of triples to the user. While the subject (or the object, for the second type, respectively) is constant, there are two choices upon which to determine semantic relatedness. We have chosen to use the predicates for determining semantic relatedness, since the resources related as subjects or objects can be quite diverse.

One further design decision was to treat both types of triples separately, For example, consider the following facts:

```
:Germany :partOf :Europe .
:Hesse :partOf :Germany .
```

Both predicates have the same name, but if we focus on the resource `Germany`, they can have different meanings. In the first line, `Germany` is subject, that means we look at a direct property. It describes that `Germany` is a part of `Europe`. The second line describes an indirect property of `Germany`, namely that `Hesse` is a part of `Germany`. If we handle both predicates as one, we could not create groups for `Location of Germany` and `States of Germany`. We want to keep this information, so we handle direct and indirect predicates as two different predicates.

2.2 Determining the Similarity of Predicates

Although predicates are defined in schemas or ontologies as object or datatype properties, those definitions carry only little information about the similarity of two predicates. For example, the two DBpedia properties `areaCode` and `leaderTitle` are defined with identical domain and range definitions, however, they are not very closely related semantically. Thus, we have to use external knowledge to determine the semantic similarity of predicates.

We use the labels and local names (if labels are not present) of properties for calculating a semantic similarity. Those are strings, which we tokenize into individual words. We then look up those words in *WordNet* [5] in order to determine their similarity. Each predicate is mapped to one or more SynSets (i.e., words of synonym meaning) in WordNet.

There are various relations between the SynSets in WordNet. Using the structure of WordNet allows us to calculate distances [4] between two words. Because semantic identical words are placed together, we can use the path length for our approach. A problem occurs if we try to calculate distances between words of different word types (e.g., nouns and verbs), because they are organized in a different SynSet, and by far not all predicates in Linked Open Data are proper verbs, as shown in the examples above. In case of a verb, we use nominalisation to produce the common subjective. All other word types will be ignored, reducing false distance calculation. When predicates consist of multiple words, we use two different kinds of metric to calculate the distance: average distance between each pair of words and the minimal distance between each pair of words.

2.3 Grouping Predicates

Once we have determined the distance between each pair of predicates viewed for a resource, we have to order them in groups. To that end, we use clustering algorithms which group those predicates, based on the computed distances. The result should be groups that contain facts with a semantic relation. For the development of the semantic browser, we have tested two cluster algorithms: k-means [10] and bottom-up [6] (hierarchical clustering).

For the k-means algorithm, we have to choose a number k how many groups should be generated by the algorithm. We have decided to use the psychological criteria: humans can only keep around 7 things in mind at the same time [8]. Thus, in our implementation the k-means generates 7 groups for every direct and indirect properties, resulting in 14 groups.

For the bottom-up algorithm, the number of groups cannot be chosen directly. The algorithm generates a tree with the grouped items, and the height at which to cut the tree can be selected. To find the best-fitting tree height, we have tested 110 random datasets:

Using the same psychological aspect [8] we chose to cut the tree at a height of 5. That results in an average number of groups of 13.20, each with 20.55 items in average. The number of items in each groups seems to be very high, but the average has the property that big datasets results in imprecise results. The

tree height	Ø number of groups	Ø group size	median number of groups	median group size
3	5.27	56.81	4.00	16.00
4	8.55	33.60	7.00	10.00
5	**13.20**	**20.55**	**12.00**	**6.50**
6	19.22	12.97	18.00	4.00
7	25.54	8.94	24.00	3.00

Table 1. Dependency between the tree height and the generated groups.

median in this case is much lower and shows that the group size is much smaller for normal-sized datasets.

2.4 Labeling Groups

Once the groups have been identified, we want to assign meaningful headlines to them. Those are displayed for structuring the document as well as for providing a shortcut navigation, as shown in Fig. 1.

We have implemented two strategies that provide group labels based on the facts that were grouped together. The first approach exploits the WordNet structure and finds common ancestors along the specialization hierarchy that matches the majority of the grouped facts' predicates. In case this is not successful (i.e., the common ancestor is a meaningless top-level concept such as *object*), the most frequent predicate's label is used as a group label.

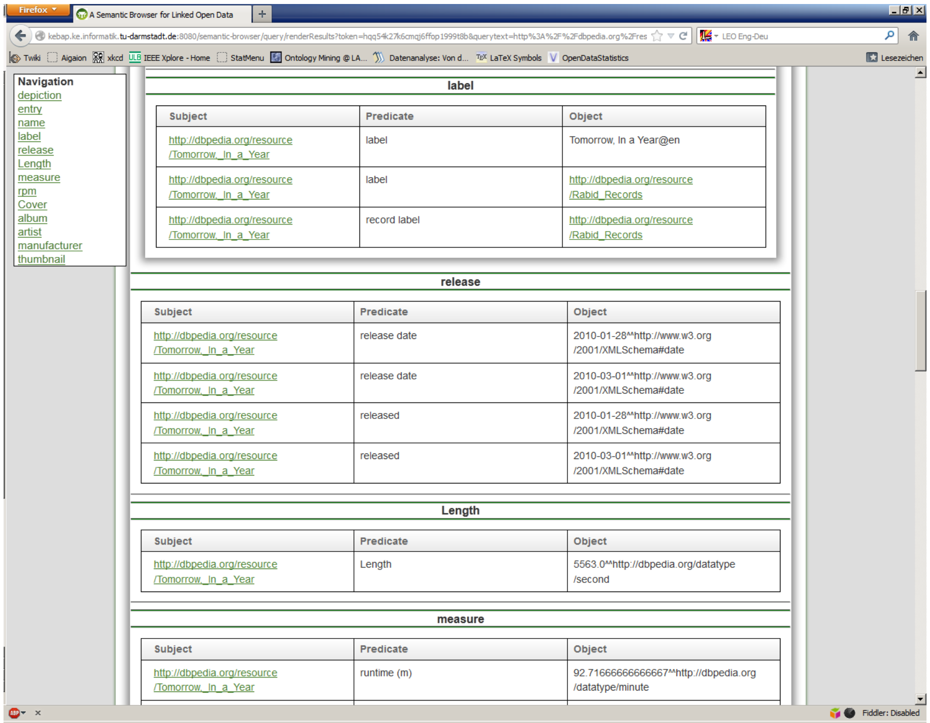
2.5 Implementation

The approach discussed above has been implemented as an extension into the modular Linked Open Data browser *MoB4LOD*[7]. In order to avoid time-consuming lookup of property labels, those labels are cached, so that after a warm-up period, most resources can be loaded and displayed within a few seconds. A screenshot of the application is shown Fig. 1.

3 Evaluation

For evaluation, a preliminary user study was performed. We randomly selected each 10 DBpedia resources from 3 different kind of topics: country, city and movie. They were selected so people of this study do not know them nor the resource has too less information for grouping properties together. For each resource a question was created manually. In the study, the user had to answer the question only by using the information that our browser provides. The idea behind this study was: the faster the user can answer the question, the better the view of the browser is. The use of the functionality to search within the page was not allowed. We measured the time to complete a question and

[7] http://www.ke.tu-darmstadt.de/resources/mob4lod/


Firefox
A Semantic Browser for Linked Open Data
kebap.ke.informatik.tu-darmstadt.de:8080/semantic-browser/query/renderResults?token=hqq54k27k6cmqj6ffop1999t8b&querytext=http%3A%2F%2Fdbpedia.org%2Fres
LEO Eng-Deu
Twiki
Aigaion
xkcd
IEEE Xplore - Home
StatMenu
Ontology Mining @ LA...
Datenanalyse: Von d...
LaTeX Symbols
OpenDataStatistics
Lesezeichen
Fiddler: Disabled

Navigation
depiction
entry
name
label
release
Length
measure
rpm
Cover
album
artist
manufacturer
thumbnail

label

Subject	Predicate	Object
http://dbpedia.org/resource /Tomorrow,_In_a_Year	label	Tomorrow, In a Year@en
http://dbpedia.org/resource /Tomorrow,_In_a_Year	label	http://dbpedia.org/resource /Rabid_Records
http://dbpedia.org/resource /Tomorrow,_In_a_Year	record label	http://dbpedia.org/resource /Rabid_Records

release

Subject	Predicate	Object
http://dbpedia.org/resource /Tomorrow,_In_a_Year	release date	2010-01-28^^http://www.w3.org /2001/XMLSchema#date
http://dbpedia.org/resource /Tomorrow,_In_a_Year	release date	2010-03-01^^http://www.w3.org /2001/XMLSchema#date
http://dbpedia.org/resource /Tomorrow,_In_a_Year	released	2010-01-28^^http://www.w3.org /2001/XMLSchema#date
http://dbpedia.org/resource /Tomorrow,_In_a_Year	released	2010-03-01^^http://www.w3.org /2001/XMLSchema#date

Length

Subject	Predicate	Object
http://dbpedia.org/resource /Tomorrow,_In_a_Year	Length	5563.0^^http://dbpedia.org/datatype /second

measure

Subject	Predicate	Object
http://dbpedia.org/resource /Tomorrow,_In_a_Year	runtime (m)	92.71666666666667^^http://dbpedia.org /datatype/minute

Fig. 1. Screenshot of the Semantic Browser Application

whether that answer was correct. Furthermore, we asked the users for feedback in a questionnaire and to rate the different views.

We created three different views for the 30 resources:

1. The *baseline* view produces an alphabetical list of the properties, grouped in eight clusters based on the first letter (A-C, D-F, etc.).
2. The clusters provided by the *k-means* clustering algorithm with $k = 7$, using the average WordNet distance for multi-word labels (see above).
3. The clusters provided by the bottom-up clustering algorithm. In this case, we have used the minimum distance for multi-word labels, since it provided better results in a pre-evaluation.

Each user was shown all 30 resources rotating the view after 10 resources, so every user had to work with all 3 views. The assignment of resources to views was shifted for each user. We asked 10 people to participate in our evaluation. In total, we have analyzed 300 answers of which only 15 were given wrong answers or the user did not find the answer. So only a small number of questions (5 %) could not be answered correctly.

The summary of results are depicted in table 2. Although there are always questions that can be solved faster with the baseline view, in total, 17 out of 30

Algorithm	Baseline	k-means	Bottom-up
avg. (median) number of groups			
Cities	5.6 (4)	12.4 (13.0)	17.9 (17.5)
Countries	5.1 (5.5)	12.0 (12.0)	17.4 (15.5)
Movies	3.2 (3)	7.3 (7.3)	11.7 (12.0)
Ø group size			
Cities	184 (37)	104.0 (12.5)	71.4 (8.0)
Countries	95.8 (22.5)	41.1 (8.0)	26.5 (6.5)
Movies	31.8 (30.5)	13.1 (12.5)	8.0 (8.0)
avg. (median) algorithm runtime (in seconds)			
Cities	< 1 (< 1)	14.1 (4.6)	15.4 (5.1)
Countries	< 1 (< 1)	3.9 (1.6)	4.3 (1.7)
Movies	< 1 (< 1)	0.5 (0.4)	0.4 (0.4)
avg. (median) user task completion time (in seconds)			
Cities	**33.6** (**25.1**)	65.0 (51.6)	40.3 (36.1)
Countries	54.5 (**40.0**)	**47.7** (42.7)	60.0 (51.1)
Movies	**33.0** (31.1)	36.1 (**27.4**)	37.0 (32.0)
User rating (1=best, 6=worst)			
	3.0	3.4	2.9

Table 2. Summary of evaluation results

questions were answered faster using one of the clustering views; 7 out of 10 for countries, 6 out of 10 for movies, 4 out of 10 for cities. In particular for questions about countries, the k-means algorithm provides the more useful view.

Another observation was that clustering did not work well on the city data. One reason is that city data in DBpedia contain many weather observations (minimum, maximum, and average temperature and rain for each month), which are mixed with other data when using the k-means algorithm, and which are grouped into many small clusters when using the bottom-up clustering algorithm. When comparing the median and the average values for the size of groups, it can be observed that there are severe outliers for city and country data, while the clustering of information about movies works much better.

In terms of the users' rating, the bottom-up clustered views are slightly preferred to the other views (while the users were faster using the k-means based view). However, none of the results are statistically significant due to the small number of users in this preliminary study.

4 Conclusion and Future Work

In this paper, we have discussed an approach for enabling *semantic* exploration of Linked Open Data. The preliminary user study has shown that it is possible to use existing linked open data together in conjunction with external linguistic resources such as WordNet to generate a more intuitive user interface for viewing RDF data sets.

There are many possible extensions of this work, from using other semantic distance measures to implementing more different clustering algorithms. Other options include clustering not only of predicates, but also of instances connected by those predicates in case there are many of such instances (for example, many people born in one city).

Acknowledgements

The authors would like to thank Chinara Mammadova, Dominik Stengler, Jan Wienand, Peter Klöckner and Melanie Weiland for their support in implementing the browser framework.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: A Nucleus for a Web of Open Data. (2007)
2. Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., , Sheets, D.: Tabulator: Exploring and Analyzing linked data on the Semantic Web. (2006)
3. Bizer, C., Gauß, T.: Disco - Hyperdata Browser: A simple browser for navigating the Semantic Web, http://www4.wiwiss.fu-berlin.de/bizer/ng4j/disco/. (2007)
4. Budanitsky, A., Hirst, G.: Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures, University of Toronto. (2001)
5. Fellbaum, C., Miller, G.: WordNet - An Electronic Lexical Database. MIT Press, Cambridge (1998)
6. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning - Data Mining, Inference, and Prediction, Second Edition. 2nd ed. 2009. corr. 3rd printing 5th printing. edn. Springer, Berlin, Heidelberg (2009)
7. Klyne, G., Carroll, J.J.: Resource description framework (rdf): Concepts and abstract syntax (2004) http://www.w3.org/TR/rdf-concepts/.
8. Miller, G.A.: The magical number seven, plus or minus two: Some limits on our capacity for processing information., Harvard University. (1956) Psychological Review.
9. Nuzzolese, A.G., Gangemi, A., Presutti, V., Ciancarini, P.: Encyclopedic Knowledge Patterns from Wikipedia Links. (2011)
10. Witten, I.H., Frank, E., Hall, M.A.: Data Mining - Practical Machine Learning Tools and Techniques. Elsevier, Amsterdam (2005)

Appendix

This appendix describes how our submission meets the challenge criteria.

Minimal Requirements

End-user Application The application is a Linked Open Data browser targeted at users of the semantic web. It provides additional value for browsing the semantic web.

Information Sources The approach is generic and works with any kind of Semantic Web data, following elementary standards such as RDF. Furthermore, WordNet is used as an external linguistic resource.

Meaning of Data The meaning of data in Linked Open Data is an essential focus of our work.

Additional Desirable Features

Web Interface The application provides a functional web interface, which we hope is attractive.

Scalability The application only works on subset of Linked Open Data (i.e., the portion that is viewed by the user). The evaluations in the paper show that for most datasets, real-time processing of the data is feasible.

Evaluation The paper contains an evaluation w.r.t. to user satisfaction.

Functionality different from or beyond pure information retrieval By providing semantic clustering of results, the interface is more than just an information retrieval interface.