4C1-IOS-4b-1

Accessing Linked Data with A Simple Integrated Ontology

Lihua Zhao Ryutaro Ichise

National Institute of Informatics, Tokyo, Japan

The Web of Data provides abundant knowledge, which contains a huge amount of data sets and links among them. In order to access to the linked data, we have to be familiar with the ontology of each data set. However, because of the heterogeneous and big ontologies, it is time consuming and difficult to manually observe which ontology schemas are important for describing a specific instance and how the same concept is represented differently in the ontologies. In order to help linked data users easily understand the ontologies and access to various data sets, we construct a global ontology by integrating related ontology schemas and by retrieving core ontology schemas. In this paper, we propose a semi-automatic ontology integration framework that reduces the heterogeneity of ontologies using ontology similarity matching on the SameAs graph patterns and retrieves frequently used core ontology schemas from the linked data with machine learning methods. This framework constructs a high-quality integrated ontology, which is easily understandable and effective in knowledge acquisition from the linked data.

1. Introduction

The Linked Open Data (LOD) is a collection of machine-readable structured data connected by *owl:sameAs*, which refers to identical instances in diverse data sets [Bizer 09]. Although a huge amount of data sets are published in the LOD cloud, there is no standard ontology for all the data sets, but all kinds of ontologies which cause the ontology heterogeneity problem. A commonly used method to overcome the ontology heterogeneity problem is the ontology matching, which finds corresponding mappings between ontologies [Pavel 13]. Since it is time-consuming and infeasible to manually inspect large ontologies, we need an automatic or semi-automatic method to integrate the heterogeneous ontologies.

In this paper, we propose a framework that semi-automatically analyzes graph patterns of interlinked instances and integrates heterogeneous ontologies by retrieving related classes and properties. We also retrieved frequently used core classes and properties that can help Semantic Web application developers easily understand the ontology schemas of the data sets. Furthermore, we enriched the integrated ontology by adding annotations, domain information, and range information that can provide us rich information about the ontology. We will show evaluation of the integrated ontology and a SPARQL query example that can discover missing links with the integrated ontology.

2. Ontology Integration Framework

The ontology integration framework integrates core ontology schemas and related ontology schemas from various data sets. Fig. 1 shows the ontology integration framework, which consists of three main components: graph-based ontology integration, machine-learning-based approach, and integrated ontology constructor [Zhao 13]. In the following, we will briefly introduce each component.

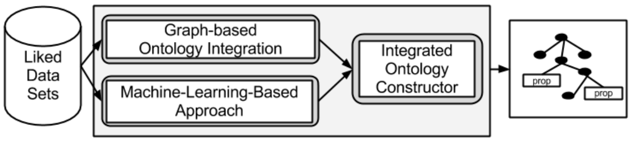
Liked Data Sets
Graph-based Ontology Integration
Machine-Learning-Based Approach
Integrated Ontology Constructor
prop
prop

Figure 1: Ontology Integration Framework.

2.1 Graph-based Ontology Integration

The instances interlinked by *owl:sameAs* constructs graphs, from which we can retrieve related classes and properties by applying ontology matching methods on the graphs [Zhao 12]. The graph-based ontology integration component contains 5 steps: graph pattern extraction, <Predicate, Object> collection, related classes and properties grouping, aggregation of all the integrated classes and properties, and manual revision. By analyzing the extracted graph patterns, we detect related classes and properties, which are classified into different data types: Date, URI, Number, and String. Similar classes are integrated by tracking the subsumption relations and different similarity matching methods are applied on different types of <Predicate, Object> pairs to retrieve similar properties. We automatically integrate related classes and properties for each graph pattern, and then aggregate all of them to construct a preliminary integrated ontology. Although we need a minor revision on the preliminary integrated ontology, it is a light work.

2.2 Machine-Learning-Based Approach

The graph-based ontology integration method can retrieve related classes and properties from different ontologies, but it may miss some core classes and frequently used properties. Therefore, we need another method to extract top-level classes and frequent core properties, which are essential for describing instances.

By applying machine learning methods, we can find frequent core properties that are essential for describing instances of a specific class. The Decision Table is a rule-based algorithm that can retrieve a subset of core proper-

Contact: 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430, lihua@nii.ac.jp, ichise@nii.ac.jp

Table 1: Evaluation of the Integrated Ontology

Data Pair	Precision	Recall	F-measure
DBpedia-Geonames	0.64	0.37	0.47
DBpedia-LinkedMDB	1	0.1	0.2
DBpedia-NYTimes	0.93	0.02	0.04
LinkedMDB-NYTimes	1	0.07	0.13
LinkedMDB-Geonames	0	0	n/a
Geonames-NYTimes	1	0.04	0.08

ties and the Apriori algorithm can find a set of associated properties that are frequently used for describing instances. Hence, we apply the Decision Table and the Apriori algorithm to retrieve top-level classes and frequent core properties from the linked data sets.

2.3 Integrated Ontology Constructor

In order to construct an easily understandable ontology, we enrich the definition of the retrieved ontology classes and properties by adding annotations, domain and range information. We automatically add these information by analyzing the values of properties from the samples of the instances in the data sets. This integrated ontology constructor mainly consists of the ontology enrichment, ontology merger, and naming validator.

3. Experiments

DBpedia, Geonames, NYTimes and LinkedMDB from the LOD cloud are used for evaluating our framework. The final integrated ontology contains 135 classes and 453 properties that are grouped into 87 and 97 groups, respectively. In this section, we evaluate the quality of the integrated ontology and evaluate the performance using the integrated ontology.

3.1 Evaluation of the Integrated Ontology

The quality of the integrated ontology is evaluated with the ontology alignments created by the experts who are familiar with the LOD data sets. They created alignments among DBpedia, Geonames, LinkedMDB, and NYTimes. The precision, recall, and F-measure of the alignments in the integrated ontology are shown in Table 1.

As shown in Table 1, the precision reaches 1 for the alignments of DBpedia-LinkedMDB, LinkedMDB-NYTimes, and Geonames-NYTimes. For the DBpedia-Geonames and DBpedia-NYTimes we found some incorrect alignments, but we could not get any alignment for the LinkedMDB-Geonames. The system can perform best to find the alignments between DBpedia and Geonames. One of the reason is that most of the links are between DBpedia and Geonames, while there are only 247 links between LinkedMDB and Geonames that are too less to find correct alignments.

3.2 Evaluation with A SPARQL Query

The integrated ontology consists of different ontology schemas from various linked data sets. Hence, we can access to various data sets with the integrated ontology classes and properties to integrate information from different data sets.

Table 2: Find Missing Links of Islands.

```
SELECT DISTINCT ?geo ?db ?string
where { ?db rdf:type db-onto:Island.
mo:name mo-prop:hasMemberProperty ?dname.
?db ?dname ?string.
?geo geo-onto:featureCode geo-onto:T.ISL.
mo:name mo-prop:hasMemberProperty ?gname.
?geo ?gname ?string. }
```

One of the advantages of using the integrated ontology is that we can find missing SameAs links.

The example in Table 2 can find missing SameAs links of the island instances between DBpedia and Geonames. The db-onto:Island and geo-onto:T.ISL are integrated in the ontology, which are used for island instances in DBpedia and Geonames, respectively. The SPARQL query finds the same islands that has the same name in DBpedia and Geonames. In total, we retrieved 509 links, among which 97 existing links are from DBpedia to Geonames, 211 links are from Geonames to DBpedia, and 90 bidirectional links between DBpedia and Geonames. Hence, we discovered 291 missing links which has the same island name.

4. Conclusion

We proposed a semi-automatic ontology integration framework that consists of three main components. The graph-based ontology integration component retrieves related ontology classes and properties by analyzing the graph patterns of the interlinked instances. The integrated ontology contains top-level classes and frequent core properties retrieved from the machine-learning-based approach, which can easily find core properties used for a specific instance. We also enriched the integrated ontology and validated it in the integrated ontology constructor component. The integrated ontology can be used to detect missing links with simple SPARQL queries.

References

[Bizer 09] Christian Bizer, Tom Heath and Tim Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.

[Pavel 13] Pavel Shvaiko and Jérôme Euzenat. Ontology Matching: State of the Art and Future Challenges. *IEEE Transactions on Knowledge and Data Engineering*, 25(1):158–176, 2013.

[Zhao 12] Lihua Zhao and Ryutaro Ichise. Graph-Based Ontology Analysis in the Linked Open Data. *Proceedings of the Eighth International Conference on Semantic Systems*, 56–63, 2012.

[Zhao 13] Lihua Zhao and Ryutaro Ichise. Instance-Based Ontological Knowledge Acquisition. *Proceedings of the Tenth Extended Semantic Web Conference*, 155–169, 2013.