TBSL Question Answering System Demo

Konrad Höffner, Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, Phillip Cimiano

Universität Leipzig, IFI/AKSW

Abstract. As an increasing amount of RDF data is published as Linked Data, intuitive ways of accessing this data become more and more important. Natural language question answering approaches have been proposed as a good compromise between intuitiveness and expressiveness. We present a user interface for the template based TBSL system which covers the full question answering pipeline and answers factual questions with a list of RDF resources. Users can ask full-sentence, English factual questions and get a list of resources which are then visualized using those properties which are expected to carry the most important information for the user. The available knowledge bases are (1) DBpedia for general domain question answering and (2) Oxford real estate for housing searches. However, the system is easily extensible to other knowledge bases.

Keywords: Question Answering, Semantic Web, Natural Language Patterns, SPARQL

1 Introduction

As more and more RDF data is published as Linked Data, developing intuitive ways of accessing this data becomes increasingly important. One of the main challenges is the development of interfaces that exploit the expressiveness of the underlying data model and query language, while hiding their complexity.

As a good compromise between intuitiveness and expressiveness, *question answering* (*QA*) approaches allow users to express arbitrarily[1] complex information needs in natural language without requiring them to be aware of the underlying schema, vocabulary or query language. Several question answering systems for RDF data have been proposed in the past, for example, Aqualog [5,7], PowerAqua [8], NLP-Reduce [2] and FREyA [1]. Many of these systems map a natural language question to a triple-based representation. For example, consider the simple question Who wrote The Neverending Story?. PowerAqua[2] would map this question to the triple representation

$$\langle \text{[person,organization]}, \text{wrote}, \text{Neverending Story} \rangle.$$

[1] At least as complex as can be represented in the query language.

[2] Accessed via the online demo at `http://poweraqua.open.ac.uk:8080/poweraqualinked/jsp/index.jsp`.

Then, by applying similarity metrics and search heuristics, it would retrieve matching subgraphs from the RDF repository. For the above query, the following triples would be retrieved from DBpedia, from which the answer "Michael Ende" can be derived:

$\langle \text{Writer}, \text{IS\_A}, \text{Person} \rangle$
$\langle \text{Writer}, \text{author}, \text{The\_Neverending\_Story} \rangle$

While this approach works very well in cases where the meaning of the query can be captured easily, it has a number of drawbacks, as in many cases the original semantic structure of the question can not be faithfully captured using triples. For instance, consider the questions 1a and 2a below. PowerAqua would produce the triple representations in 1b and 2b, respectively. The goal, however, would be SPARQL queries[3] like 1c and 2c, respectively.

1. (a) Which cities have more than three universities?
   (b) $\langle [\text{cities}], \text{more than}, \text{universities three} \rangle$
   (c)
```
SELECT ?y WHERE {
   ?x rdf:type onto:University .
   ?x onto:city ?y .
}
HAVING (COUNT(?x) > 3)
```
2. (a) Who produced the most films?
   (b) $\langle [\text{person,organization}], \text{produced}, \text{most films} \rangle$
   (c)
```
SELECT ?y WHERE {
   ?x rdf:type onto:Film .
   ?x onto:producer ?y .
}
ORDER BY DESC(COUNT(?x)) OFFSET 0 LIMIT 1
```

Such SPARQL queries are difficult to construct on the basis of the above mentioned triple representations, as aggregation and filter constructs arising from the use of specific quantifiers are not faithfully captured. What would be needed instead is a representation of the information need that is much closer to the semantic structure of the original question. Thus, the TBSL approach to question answering over RDF data relies on a parse of the question to produce a SPARQL template that directly mirrors the internal structure of the question and that, in a second step, is instantiated by mapping the occurring natural language expressions to the domain vocabulary. For example, a template produced for Question 2a would be:

3.
```
SELECT ?x WHERE {
   ?x ?p ?y .
   ?y rdf:type ?c .
}
ORDER BY DESC(COUNT(?y)) LIMIT 1 OFFSET 0
```

[3] Assuming a DBpedia namespace with `onto` as prefix `<http://dbpedia.org/ontology/>`.

In this template, `c` stands proxy for the URI of a class matching the input keyword films and `p` stands proxy for a property matching the input keyword produced. In a next step, `c` has to be instantiated by a matching class, in the case of using DBpedia `onto:Film`, and `p` has to be instantiated with a matching property, in this case `onto:producer`. For instantiation, we exploit an index as well as a pattern library that links properties with natural language predicates.

This approach is shown [6] to be competitive and specific cases of questions that can be precisely answered with this approach but not with competing approaches are discussed there. This is done by evaluating it against the QALD benchmark, where of the 50 questions, 34 are manually determined to be solvable and 19 are answered fully correctly.

The main contribution of this paper is to provide a practical proof-of-concept of the TBSL methodology and its flexibility in adapting to different knowledge bases.

In the online demo, users can choose between two different knowledge bases: DBpedia [3] is the main information hub of the semantic web and contains factual information about several million resources and hundreds of millions of facts about them. The Oxford real estate knowledge base [4] contains relevant information for potential house-buyers in the Oxford area, such as the price, location, number of bedrooms and bathrooms as well as images and textual descriptions.

In the following section we present an overview of the system's architecture, in Section 3, we describe the online demo and finally we elaborate about future work.

2 Architecture

Figure 1 gives an overview of our approach. The input question, formulated by the user in natural language, is first processed by a POS tagger. On the basis of the POS tags, lexical entries are created using a set of heuristics. These lexical entries, together with pre-defined domain-independent lexical entries, are used for parsing, which leads to a semantic representation of the natural language query, which is then converted into a SPARQL query template. The query templates contain *slots*, which are missing elements of the query that have to be filled with URIs. In order to fill them, our approach first generates natural language expressions for possible slot fillers from the user question using WordNet expansion. In a next step, sophisticated entity identification approaches are used to obtain URIs for those natural language expressions. These approaches rely both on string similarity as well as on natural language patterns which are compiled from existing structured data in the Linked Data cloud and text documents. This yields a range of different query candidates as potential translations of the input question. It is therefore important to rank those query candidates. To do this, we combine string similarity values, prominence values and schema conformance checks into a score value. The highest ranked queries are then tested against the

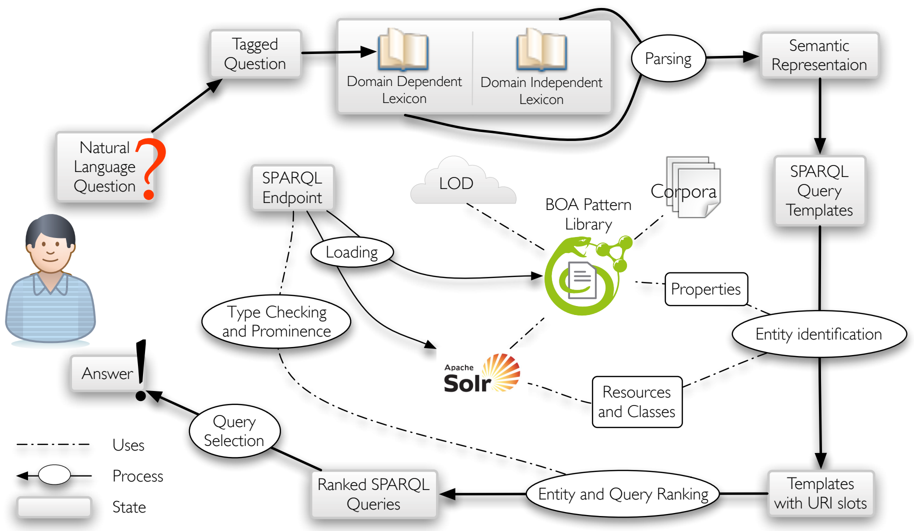


Fig. 1. Overview of the template based SPARQL query generator.

underlying triple store and the best answer is returned to the user. The approach is explained in more detail in the original publication [6].

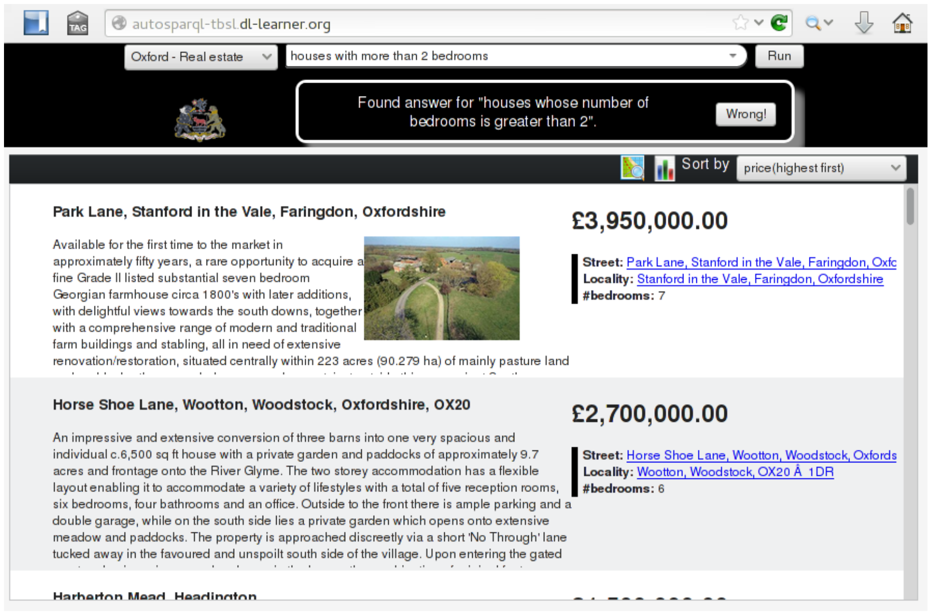
3 Demo Description

The online demo can be found at `http://autosparql-tbsl.dl-learner.org/`. As the users enters a question in the search field, the list of answer resources is visualized, see Figure 2. As the answer type of our system is a list of RDF resources, only factual questions which can be answered by such a list can be answered. Examples of unanswerable queries are “How old is Stephen King?”, “Why is the sky blue?” and “Is the pope catholic?”.

Additional Functionality In case the question is wrongly interpreted, expert users can choose among different interpretations, see Figure 3. When using the Oxford knowledge base, there are three additional features: (1) the resources can be sorted by criteria such as the price or number of rooms, (2) they can be shown according to their price in a bar graph and (3) they can be displayed in a map.

Identifying Relevant Information While for Oxford houses the set of properties is fixed, in DBpedia, a RDF resource can have a big number of properties which leads to a very wide row in our table layout and makes it hard for a user to find the relevant information. To shrink the number of properties visible by default, those properties are internally sorted by their frequency in the answer set and are filtered by first applying an absolute threshold and then selecting only the k

properties with the highest frequencies. Furthermore there is a manual blacklist that contains properties which relate to Wikipedia or DBpedia articles instead of the resources they represent and are typically not relevant for a user in a question answering context, such as `wikiPageUsesTemplate`. If a users wishes to add a property which is not included in the default selection, the property can always be added by selecting it in the "Show also" box.


autosparql-tbsl.dl-learner.org
Oxford - Real estate
houses with more than 2 bedrooms
Run
Wrong!
Sort by
price(highest first)

Found answer for "houses whose number of bedrooms is greater than 2".

Park Lane, Stanford in the Vale, Faringdon, Oxfordshire

£3,950,000.00

Available for the first time to the market in approximately fifty years, a rare opportunity to acquire a fine Grade II listed substantial seven bedroom Georgian farmhouse circa 1800's with later additions, with delightful views towards the south downs, together with a comprehensive range of modern and traditional farm buildings and stabling, all in need of extensive renovation/restoration, situated centrally within 223 acres (90.279 ha) of mainly pasture land

Street: Park Lane, Stanford in the Vale, Faringdon, Oxfc
Locality: Stanford in the Vale, Faringdon, Oxfordshire
#bedrooms: 7

Horse Shoe Lane, Wootton, Woodstock, Oxfordshire, OX20

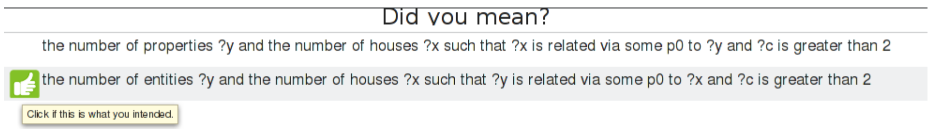
£2,700,000.00

An impressive and extensive conversion of three barns into one very spacious and individual c.6,500 sq ft house with a private garden and paddocks of approximately 9.7 acres and frontage onto the River Glyme. The two storey accommodation has a flexible layout enabling it to accommodate a variety of lifestyles with a total of five reception rooms, six bedrooms, four bathrooms and an office. Outside to the front there is ample parking and a double garage, while on the south side lies a private garden which opens onto extensive meadow and paddocks. The property is approached discreetly via a short 'No Through' lane tucked away in the favoured and unspoilt south side of the village. Upon entering the gated

Street: Horse Shoe Lane, Wootton, Woodstock, Oxfords
Locality: Wootton, Woodstock, OX20 Â 1DR
#bedrooms: 6

Harberton Mead, Headington

Fig. 2. The question "houses with more than 2 bedrooms" on the Oxford real estate knowledge base is answered by the with a list of houses and their properties.



Did you mean?

the number of properties ?y and the number of houses ?x such that ?x is related via some p0 to ?y and ?c is greater than 2

the number of entities ?y and the number of houses ?x such that ?y is related via some p0 to ?x and ?c is greater than 2

Fig. 3. Expert users can choose among different interpretations for difficult questions.

Implementation The demo is implemented in Java to make use of the rich library support for both *natural language processing* (*NLP*) and RDF. It uses the *Apache Jena*[4] framework which provides RDF and SPARQL capabilities and

[4] `jena.apache.org`

the Stanford JavaNLP API[5] for NLP functions such as part-of-speech tagging. The user interface is a web application[6] and is implemented in *Vaadin*[7], which is a server-centric extension to the *Google Web Toolkit*[8]. The DBpedia knowledge base is accessed via the default SPARQL endpoint[9] while the much smaller Oxford housing data is loaded into a memory model from a local RDF dump. An Apache Solr[10] index is used to get resource candidates for a template slot from the DBpedia and Oxford knowledge bases. The inverted index provides an efficient lookup from template slots, like "book" in "Give me all books written by Dan Brown", to resources like `http://dbpedia.org/class/Book` which contain this word in its values for the `rdfs:label` or `rdfs:comment` property, including near matches.

4 Future Work

We are planning improvements for both the user interface and the core algorithm.

4.1 User Interface

User Feedback We plan to increase the amount of feedback the user gets, especially regarding his search query. For time consuming queries, an estimate tells the user, that the program has not crashed but is still working and helps identify, whether the users' information needs are worth the waiting time. A detailed explanation of errors, for example that a "why-query" is generally not possible but that another one only fails, because the wanted information is not contained in the knowledge base, helps the user in formulating further queries. Furthermore, we plan to persist feedback that the program gets from the user and learn from it, like the rejection of the default interpretation and the choice of an alternative.

Use of different Algorithms We plan to allow using additional QA algorithms so that users can choose the algorithm that is best suited to the task at hand.

Performance and Usability Evaluation While the quality of the TBSL algorithm has already been evaluated using the QALD benchmark (see section 1), an evaluation of user interface has yet to be done. We plan to do a quantitative study where we measure the time spent between the user entering a query and the complete displaying of the results. Furthermore we plan a qualitative study where users are asked to freely give feedback about their expectations about the interface and their experiences with it.

[5] `http://www-nlp.stanford.edu/software/`
[6] `http://autosparql-tbsl.dl-learner.org/`
[7] `https://vaadin.com`
[8] `http://www.gwtproject.org`
[9] `http://dbpedia.org/sparql`
[10] `http://lucene.apache.org/solr/`

4.2 Algorithm

We are investigating into several qualitative and quantitative improvements of the TBSL algorithm.

Optimizing the URI Disambiguation It is crucial to find the right resource for each slot because even a failure on a single one misrepresents the information need of the query and produces answers which are useless to the user. As such we strive to achieve an disambiguation rate that is as high as possible.

Multilingualism Natural language question answering aims to remove barriers to the access of Semantic Web data by allowing lay users to query it in their own language instead of a synthetic query language like SPARQL. Adding more natural languages further increases the potential user base and facilitates querying for users who can speak English but do not have it as a native language. At first we plan to add the German language by creating German patterns and a German domain-independent dictionary.

References

1. H. Cunningham D. Damljanovic, M. Agatonovic. Natural language interfaces to ontologies: Combining syntactic analysis and ontology-based lookup through the user interaction. In *Proceedings of the 7th Extended Semantic Web Conference (ESWC 2010), Heraklion, Greece, May 31-June 3, 2010*. Springer, 2010.
2. L. Fischer E. Kaufmann, A. Bernstein. NLP-Reduce: A "naive" but domain-independent natural language interface for querying ontologies. In *Proceedings of the 4th European Semantic Web Conference (ESWC 2007), Innsbruck, Austria*, 2007.
3. J. Lehmann, C. Bizer, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. DBpedia – A crystallization point for the Web of Data. *Journal of Web Semantics*, 7(3):154–165, 2009.
4. Jens Lehmann, Tim Furche, Giovanni Grasso, Axel-Cyrille Ngonga Ngomo, Christian Schallhart, Andrew Sellers, Christina Unger, Lorenz Bühmann, Daniel Gerber, Konrad Höffner, David Liu, and Sören Auer. Deqa: Deep web extraction for question answering. In *Proceedings of ISWC*, 2012.
5. V. Lopez and E. Motta. Ontology driven question answering in AquaLog. In *Proceedings of the 9th International Conference on Applications of Natural Language to Information Systems (NLDB 2004), Manchester, England*, 2004.
6. Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. Sparql template-based question answering. In *Proceedings of WWW*, 2012.
7. E. Motta V. Lopez, V. Uren and M. Pasin. AquaLog: An ontology-driven question answering system for organizational semantic intranets. *Journal of Web Semantics*, 5(2):72–105, 2007.
8. V. Uren V. Lopez, M. Sabou and E. Motta. Cross-ontology question answering on the Semantic Web – an initial evaluation. In *Proceedings of the Knowledge Capture Conference, 2009, California*, 2009.