Evaluation of SPARQL query generation from natural language questions

K. Bretonnel Cohen
Computational Bioscience Program
U. Colorado School of Medicine

Jin-Dong Kim
Database Center for Life Science

Abstract

SPARQL queries have become the standard for querying linked open data knowledge bases, but SPARQL query construction can be challenging and time-consuming even for experts. SPARQL query generation from natural language questions is an attractive modality for interfacing with LOD. However, how to evaluate SPARQL query generation from natural language questions is a mostly open research question. This paper presents some issues that arise in SPARQL query generation from natural language, a test suite for evaluating performance with respect to these issues, and a case study in evaluating a system for SPARQL query generation from natural language questions.

1 Introduction

The SPARQL query language is the standard for retrieving linked open data from triple stores. SPARQL is powerful, flexible, and allows the use of RDF, with all of its advantages over traditional databases. However, SPARQL query construction has been described as "absurdly difficult" (McCarthy et al., 2012), and even experienced users may struggle with it. For this reason, various methods have been suggested for aiding in SPARQL query generation, including assisted query construction (McCarthy et al., 2012) and, most germaine to this work, converting natural language questions into SPARQL queries.

Although a body of work on SPARQL query generation from natural language questions has been growing, no consensus has yet developed about how to evaluate such systems. (Abacha and Zweigenbaum, 2012) evaluated their system by manual inspection of the SPARQL queries that they generated. No gold standard was prepared—the authors examined each query and determined whether or not it accurately represented the original natural language question. (Yahya et al., 2012) used two human judges to manually examine the output of their system at three points—disambiguation, SPARQL query construction, and the answers returned. If the judges disagreed, a third judge examined the output. (McCarthy et al., 2012) does not have a formal evaluation, but rather gives two examples of the output of the SPARQL Assist system. (This is not a system for query generation from natural language questions per se, but rather an application for assisting in query constructions through methods like auto-completion suggestions.) (Unger et al., 2012) is evaluated on the basis of a gold standard of answers from a static data set. It is not clear how (Lopez et al., 2007) is evaluated, although they give a nice classification of error types. Reviewing this body of work, the trends that have characterized most past work are that either systems are not formally evaluated, or they are evaluated in a functional, black-box fashion, examining the mapping between inputs and one of two types of outputs—either the SPARQL queries themselves, or the answers returned by the SPARQL queries. The significance of the work reported here is that it attempts to develop a unified methodology for evaluating systems for SPARQL query generation from natural language questions that meets a variety of desiderata for such a methodology and that is generalizable to other systems besides our own.

In the development of our system for SPARQL query generation from natural language questions, it became clear that we needed a robust approach to system evaluation. The approach needed to meet a number of desiderata:

- Automatability: It should be possible to automate tests so that they can be run automatically many times during the day and so that there is no opportunity for humans to miss errors when doing manual examination.

- Granularity: The approach should allow for granular evaluation of behavior—that is, rather than (or in addition to) just returning a single metric that characterizes performance over an entire data set, such as accuracy, it should allow for evaluation of functionality over specific types of inputs.
- Modularity: The approach should allow for evaluating individual modules of the system independently.
- Functionality: The approach should allow functional, black-box evaluation of the end-to-end performance of the system as a whole.

The hypothesis being explored in the work reported here is that it is possible to conduct a principled fine-grained evaluation of software for SPARQL query generation from natural language questions that is effective in uncovering weaknesses in the software.

As in any software testing situation, various methods of evaluating the software exist. A typical black-box approach would be to establish a gold standard of the SPARQL queries themselves, and/or of the answers that should be returned in response to a natural language question.. However, we ruled out applying the black-box approach to the SPARQL queries themselves because there are multiple correct SPARQL queries that are equivalent in terms of the triples that they will return from a linked open data source. We ruled out a black-box approach based entirely on examining the triples returned from the query when the SPARQL query was executed against the triple store because the specific list of triples is subject to change unpredictably as the contents of the triple store are updated by the data maintainers.

We opted for a gray-box approach, in which we examine the output at multiple stages of processing. The first was at the point of mapping to TUIs. The Unified Medical Language System's Semantic Network contains a hierarchically grouped set of 133 semantic types, each with a Type Unique Identifier (TUI). That is, for any given natural language question that should cause a mapping to a TUI, we examined if a TUI was generated by the system and, if so, if it was the correct TUI. The second was the point of SPARQL query generation, where we focused on syntactic validity, rather than the entire SPARQL query (for the reason given above). We also examined the output of the SPARQL query, but not in terms of exact match to a gold standard. In practice, the queries would typically return a long list of triples, and the specific list of triples is subject to change unpredictably as the contents of the triple store are updated by the OMIM maintainers. For that reason, we have focused on ensuring that we know one correct triple which should occur in the output, and validating the presence of that triple in the output. We have also inspected the output for triples that we knew from domain expertise should not be returned, although we have done that manually so far and have not formalized it in the test suite.

In this paper, we focus on one specific aspect of the gray-box evaluation: the mapping to TUIs. As will be seen, mapping to TUIs when appropriate, and of course to the correct TUI, is an important feature of answering domain-specific questions. As we developed our system beyond the initial prototype, it quickly became apparent that there was a necessity to differentiate between elements of the question that referred to specific entities in the triple store, and elements of the question that referred to general semantic categories. For example, for queries like *What genes are related to heart disease?*, we noticed that *heart disease* was being mapped to the correct entity in the triple store, but *genes*, rather than being treated as a general category, was also being mapped (erroneously) to a particular instance in the triple store. Given the predicates in the triple store, the best solution was to recognize general categories in questions and map them to TUIs. Therefore, we developed a method to recognize general categories in questions and map them to TUIs. Testing this functionality is the main topic of this paper.

2 Materials and methods

2.1 Online Mendelian Inheritance In Man

In this work we focused on a single linked open data source, known as Online Mendelian Inheritance in Man (OMIM) (Amberger et al., 2011).

The most obvious application of OMIM, and the one that biomedical researchers are most accustomed to using it for, is queries about genes and diseases, but this is a **much** richer resource that is probably not often exploited to the full extent that it could be; in fact, the web-based interface offers no options at all for exploiting it beyond querying for genes and diseases.

The knowledge model goes far beyond this. It includes linkages between at least 12 semantic types, listed below in the Results section. OMIM makes use of TUIs in typing the participants in many of the triples that it encodes. In particular, each of the linkages described above is actually a pair of TUIs.

2.2 LODQA

To understand the evaluation methodology that we developed, it is helpful to understand the system under test. A prototype version of the system that differed from the current system primarily in terms of not performing TUI identification and of using a default relation for all predicates is described in some detail in (Kim and Cohen, 2013). We briefly describe the current version of the system here.

2.2.1 Architecture

In order both to understand what features of our system need to be tested and to understand how well the testing approach will generalize to evaluating other systems for SPARQL query generation from natural language questions, it is helpful to understand, in general terms, the architecture of the system that we are testing. The primary modules of the system are as follows:

- A dependency parser for determining semantic relations in the question.
- A base noun chunker for finding terms that need to be mapped to entities or TUIs in the linked open data set.
- A system for matching base noun chunks to entities or TUIs in the linked open data set.
- A module for presudo-SPARQL generation.
- A module for generating the final SPARQL query.

The user enters a question in natural language and selects a linked open data source to be queried. The first step is production of a dependency parse, using a version of the Enju parser that was trained on a corpus of English-language questions. This parse is used to identify the "target" of the question, e.g. the *wh*-NP *genes* in *What genes are associated with heart failure?* Base noun chunks are then extracted using a pattern-based approach. A pseudo-SPARQL query is generated, using the base noun chunks. TUIs are identified. The REST API of the OntoFinder system is then used to identify entities from the relevant linked open data source. For example, with OMIM selected as the linked open data source, in the query *What genes are associated with congestive heart failure?*, the base noun chunk *genes* is mapped to the TUI *T028* and the base noun chunk *congestive heart failure* is mapped to the OMIM entity *MTHU005753*. Finally, the full SPARQL query is generated, including the appropriate relation type.

2.3 A test suite for LODQA and OMIM

To build the test suite, we used three strategies: exploring a single disease in depth, sampling a variety of topics from a broad domain, and exploring linguistic variability. We began by selecting a disease represented in OMIM that had links to a wide variety of semantic types—Kearns-Sayre syndrome, a syndrome associated with ophthalmoplegia, pigmentary degeneration of the retina, and cardiomyopathy, caused by mitochondrial deletions (Kearns, 1965). For a broad domain, we chose cardiology, since one of the authors has extensive experience in that clinical area.

We accessed the entry for Kearns-Sayre syndrome through the National Center for Biomedical Ontology BioPortal. To obtain a representative sample of TUIs, we exhaustively followed all links in the RO field. We then constructed at least one pair of questions for each pair of semantic types—one that required recognizing the *type* of the TUI, and one that required recognizing an *instance* of an entity with that TUI. For example, for the TUI *Congenital Abnormality*, we constructed the questions *What congenital abnormalities are associated with Kearns-Sayre syndrome?* (type) and *What diseases is microcephaly associated with?* (where *microcephaly* is an entity of type *Congenital Abnormality*). We then explored the CHD field, yielding two additional TUIs. 10 of the TUIs could reasonably be expected to occur in natural questions (details below), and we built questions for all of these.

3 Results

The strategy described above resulted in a set of 38 natural language questions. Since the point of a test suite is to uncover problems, we focus here on the ability of the test suite to find problems, rather than focusing on how many tests we passed.

Within the set of triples for Kearns-Sayre syndrome, we found the following types of relations:

- Disease or Syndrome → Anatomical Abnormality
- Disease or Syndrome → Finding
- Disease or Syndrome → Functional Concept
- Disease or Syndrome → Disease or Syndrome
- Disease or Syndrome → Sign or Symptom
- Disease or Syndrome → Congenital Abnormality
- Disease or Syndrome → Mental or Behavioral Dysfunction
- Disease or Syndrome → Cell or Molecular Dysfunction
- Disease or Syndrome → Body Part, Organ, or Organ Component
- Disease or Syndrome → Clinical Attribute
- Disease or Syndrome → Qualitative Concept
- Disease or Syndrome → Pathologic Function

Of these twelve TUIs, 10 had forms from which one could produce a natural-sounding natural language question, the exceptions being *Functional Concept* and *Qualitative Concept*. Out of 10 TUIs that we tested, we were able to find problems with 3 of them. The problems ranged from failures to recognize TUIs to crashing the system.

Although there is as yet no non-application-specific taxonomy of error types in the task of SPARQL query generation from natural language questions (although see (Lopez et al., 2007) for an application-specific one), a number of categories that should appear in such a taxonomy presented themselves in the results of our tests:

- Number disagreements between the canonical form in the triple store and the most natural form in natural language. For example, concepts most commonly occur in the names of TUIs in singular form, e.g. *Gene or Genome*, but are more likely to occur in natural language questions in the plural, e.g. *What genes are associated with cystic fibrosis?*
- Various forms of conjunctive and disjunctive coordination, often with the potential for scope ambiguity. The most natural form of the TUI *Sign or Symptom* in a natural language question is *What signs* ***and*** *symptoms*, and this form caused the application to crash.
- Discrepancies between part of speech in the canonical form in the triple store and the most natural form in natural language. For example, the TUI *Cell or molecular dysfunction* is more likely to appear in natural language as *cellular dysfunction* than as *cell dysfunction*; these more natural forms were not recognized.

4 Discussion and conclusions

Initial exploration of a linked open data source is likely to reveal issues that must be resolved in order for natural language processing techniques to succeed. In our case, an immediate challenge that presented itself was recognizing when noun phrases should be mapped to TUIs, rather than being mapped to entities in the knowledge base.

Having recognized an issue to be resolved, test suites can be constructed to evaluate how well the issue is handled. In our case, this meant determining the full range of interacting TUIs in the knowledge base through manual exploration of the data, and then constructing a test suite that contained strings that should map to these TUIs, with the full range of linguistic variation that they can manifest.

A reasonable question to ask is whether this methodology is generalizable. Although it remains to be demonstrated, we suspect that it is, in two directions. The first is generalization to other linked open data sources. It seems likely that the methodology will generalize to the many biomedical linked open data sources in the National Center for Biomedical Ontology BioPortal. The second is generalization to other systems for SPARQL query generation from natural language questions. Based on a review of related work, we suspect that the methodology could be applied with very little adaptation to the systems described in (Lopez et al., 2007), (Abacha and Zweigenbaum, 2012), and (Yahya et al., 2012), which all have architectures that lend themselves to the type of gray-box evaluation that we have done here.

Future work will extend in a variety of directions. Following the test-driven development paradigm, further construction of the test suite will proceed ahead of the development of the LODQA application. Pressing issues for future development as we evolve further in the continuum from prototype proof-of-concept to fully-functioning application are automatic determination of the relation between the subject and object; handling negation; and handling questions that require querying multiple triple stores.

References

Asma Ben Abacha and Pierre Zweigenbaum. 2012. Medical question answering: translating medical questions into SPARQL queries. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, pages 41–50.

Joanna Amberger, Carol Bocchini, and Ada Hamosh. 2011. A new face and new challenges for Online Mendelian Inheritance in Man (OMIM®). *Human Mutation*, 32(5):564–567.

T.P. Kearns. 1965. External ophthalmoplegia, pigmentary degeneration of the retina, and cardiomyopathy: a newly recognized syndrome. *Transactions of the Ophthalmological Society of the United Kingdom*, 63:559–625.

Jin-Dong Kim and K.B̃retonnel Cohen. 2013. Natural language query processing for SPARQL generation: A prototype system for SNOMED CT. In *Proceedings of BioLINK 2013: Roles for text mining in biomedical knowledge discovery and translational medicine*, pages 32–38.

Vanessa Lopez, Victoria Uren, Enrico Motta, and Michele Pasin. 2007. AquaLog: An ontology-driven question answering system for organizational semantic intranets. *Journal of Web Semantics*.

Luke McCarthy, Ben Vandervalk, and Mark Wilkinson. 2012. SPARQL Assist language-neutral query composer. *BMC Bioinformatics*, 13.

Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. 2012. Template-based question answering over RDF data. In *Proceedings of the 21st international conference on World Wide Web*, pages 639–648.

Mohamed Yahya, Klaus Berberich, Shady Elbassuoni, Maya Ramanath, Volker Tresp, and Gerhard Weikum. 2012. Natural language questions for the web of data. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 379–390.