Generating and Executing Complex Natural Language Queries across Linked Data

Thierry Hamon[a], Fleur Mougin[b], Natalia Grabar[c]

[a]*LIMSI-CNRS, Orsay, France; Université Paris 13, Sorbonne Paris Cité, France; thierry.hamon@limsi.fr*
[b]*Université Bordeaux, ISPED, Centre INSERM U897, ERIAS, France; fleur.mougin@isped.u-bordeaux2.fr*
[c]*STL UMR8163 CNRS, Université Lille 3, France; natalia.grabar@univ-lille3.fr*

Abstract

With the recent and intensive research in the biomedical area, the knowledge accumulated is disseminated through various knowledge bases. The combined exploitation of this knowledge is important, which requires to create links between these bases and to use them jointly. Linked Data, SPARQL language and interfaces in Natural Language question-answering provide interesting solutions for querying such knowledge bases. We propose a method for translating natural language questions in SPARQL queries. We use Natural Language Processing tools, semantic resources and the RDF triples description. The method is designed on 50 questions over 3 biomedical knowledge bases, and evaluated on 27 questions. It achieves 0.78 F-measure on the test set. The method for translating natural language questions into SPARQL queries is implemented as Perl module available at http://search.cpan.org/~thhamon/RDF-NLP-SPARQLQuery

Keywords:

Natural Language Processing, SPARQL, Linked Data, biomedical domain, semantic resources, frames, Knowledge Databases

Introduction

The knowledge bases (KBs) that record existing biomedical knowledge are becoming increasingly available, although they are spread over different life-science bases that usually focus on a specific data type: chemical, pharmacological and target information on drugs in Drugbank [1], clinical studies [2], drugs and their side effects in Sider [3], etc. Creation of connections between these knowledge bases is an important research question [4]. This is indeed crucial for obtaining more exhaustive view and is also required for producing new knowledge from the already available data. The knowledge encoded in the KBs and dataset interlinks are usually represented as RDF triples, on the basis of which the linked data can be queried through a SPARQL end-point. However, it remains difficult for typical users of such knowledge sources (mainly physicians and life-science researchers) to manage the syntactic and semantic specificities of the SPARQL language together with the structure of various KBs. In order to be able to efficiently use such KBs it is necessary to design friendly interfaces that mediate the technical and semantic complexity of the task, and to provide simpler approaches for querying the KBs. It has been shown that natural language interfaces may provide the best solution [5]. The main challenge is then to propose efficient methodologies for an easy and reproducible rewriting of natural language questions in SPARQL queries.

We present a method for transforming natural language questions into SPARQL queries, and their application to biomedical Linked Data. The method is based on the use of Natural Language Processing (NLP) methods and semantic resources for enriching questions with specific annotations. Questions are then translated in SPARQL with a rule-based approach. For designing the proposed approach, we use questions proposed by the QALD challenge task 2 [6] and additional questions for evaluation. We work with three KBs (Drugbank, Diseasome, and Sider), and can address questions related for instance to drug composition and properties, to the related disorders, and to adverse effects of drugs. We first present the related work. We describe the proposed method and semantic resources available and developed for processing and translating the questions. We then present the results obtained and their evaluation.

Related work

It is possible to distinguish two kinds of objectives: design methods that transform natural language questions in SPARQL queries [7-8], and design methods that perform this transformation and also execute the queries over Linked Data. In both cases, it is necessary to define the end user interface which hides the underlying structure of the knowledge bases and the SPARQL syntax. We are interested to investigate the second purpose, which also provides the possibility to properly evaluate the methods and the results obtained. Three possibilities are usually explored for querying the Linked Data [8]: Knowledge-Based Specific Interface, Graphical Query Builder and Question-Answering System. It has also been demonstrated that, for this task, the use of natural language is preferred to the use of keywords, menus or graphs [6]. Among the existing work, we can mention the use of Question-Answering systems on 50 questions from DBpedia proposed by the QALD-2 challenge, that gives 0.62 average F-measure obtained with 39 questions. Average recall for these 39 questions is 0.63 and average precision is 0.61, although 11 questions cannot be covered by the templates [9]. Works on exploring KBs, that describe the general language, are frequent [10-11] , although there are some works on processing KBs from biomedical area. For instance, experiments have been proposed on translation of medical questions from a journal into SPARQL queries. This work combines the SVM machine learning-based approach with patterns to generate the SPARQL queries. The evaluation is carried out on 100 questions and the corresponding queries are tested on clinical documents. Method achieves 0.62 precision [12].

Material

Questions and reference data are provided by the Question Answering over Linked Data (QALD) challenge 2014 [13] dedicated to retrieval of biomedical information in linked

knowledge bases with questions in natural language. This question set is composed of 50 questions which we use for setting up the methods. We create 27 additional questions for testing the method. Example of the questions processed:

Which foods does fluvoxamine interact with?

Are there drugs that target the Probable arabinosyltransferase A?

Which genes are associated with subtypes of rheumatoid arthritis?

Which disease has the highest degree?

Which targets are involved in immune function?

Our method relies on existing biomedical resources with semantic entities data, and on additional resources collected and built for supporting the method. The steps of the method are described in the next section.

Domain-specific Resources. To process the question set, we use three biomedical resources:

- Drugbank http://www.drugbank.ca is dedicated to drugs [1]. It merges chemical, pharmacological and pharmaceutical information from other available biomedical KBs. We exploit the documentation [14] of this resource to define the rewriting rules and regular expressions for the named entity recognition.
- Diseasome [15] is dedicated to diseases and genes linked among them by known disorder/gene associations [16]. It provides a single framework with all known phenotypes and disease gene associations, indicating the common genetic origin of many diseases. We exploit the RDF triples and documentation of the resource to define the rewriting rules.
- Sider [17] is dedicated to adverse drug effects [18]. It contains information on marketed medicines and their recorded adverse drug reactions. Information is extracted from public documents and package inserts, and provides: side effect frequency, drug and side effect classifications, links to other data, such as drug-target relations.

The content of each resource is provided in specific format: RDF triples *subject predicate object*. For this reason, we can exploit their RDF schema to define frames.

Additional Resources for Question Annotation. On the basis of the RDF triples, frames are built from the RDF schemas in which the RDF predicate is the frame predicate, and subject and object of the RDF triples are the frame elements. This also includes the OWL *sameAs* triples. Several types of frame entities are isolated:

- Subject, object and predicate from triples become semantic entities. They may occur in questions: in this way, the frames are the main resource for rewriting questions in queries;
- The vocabulary specific to questions is also built. It covers for instance aggregation and negation operators , and types of questions;
- RDF literals, issued from named entity recognizer or term extractor, complete the resources. RDF literals are detected with specifically designed automata that rely on the source KB documentation.

These entities are associated with the expected semantic types, which allows creating the queries and rewriting the RDF triples in the SPARQL queries. In that respect, we can consider IRI (internationalized resource identifier), strings, common datatype or regular expressions when literals are expected. Most of the entities are treated through their semantic types, although some ambiguous entities (e.g. interaction or class) are treated atomically. For these, the rewriting rules are applied contextually to generate the semantic entities corresponding to the frames. During the query construction step, semantic types become variables and are used for connecting the edges of queries.

Methods

The method is rule-based and is composed of five parts: linguistic and semantic annotation of questions, question abstraction, query construction, query execution, and evaluation. The main steps of the method are exemplified on the question *Give me drugs in the gaseous state*. The method is based on NLP tools and methods, and must respect constrains of the SPARQL syntax.

Linguistic and Semantic Annotation of Questions is performed in several steps: identification of numerical values (corresponding to numbers or solubility index for instance) with a named entity recognizer; parsing of questions in words; part-of-speech tagging and lemmatization of questions with TreeTagger [19] (Figure 1); annotation of semantic entities (terms and their associated semantic types) with the TermTagger Perl module [20] and the semantic resources like disease names and side effects (Figure 2). In order to complete the coverage of semantic resources, we use the term extractor YaTeA [21-22] for the identification of noun phrases.

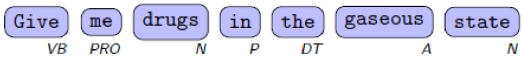

Figure 1: Linguistic annotation of questions.

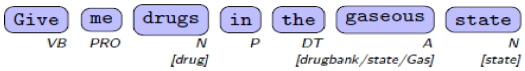

Figure 2: Semantic annotation of questions.

Question Abstraction. The objectives of the question abstraction step is to identify relevant elements within questions and to build a representation of these elements (Figure 3). This step relies on linguistic and semantic annotations. For instance, the word *drug* receives the semantic type *Drugs* from Drugbank. The main difficulty is that some entities may receive conflicting or concurrent semantic types. The purpose is then to select those semantic types that are the most correct and useful for the next steps. This is done as follows:

- We keep larger terms which do not include other semantic entities;
- We define and apply rewriting rules on the training set in order to modify or delete semantic types associated with a given entity according to the context;
- We can also modify or delete entities according to their context. For instance, the semantic entity *interaction* must be rewritten in *interactionDrug1* if the mention of drug is found in its context, but it has to be rewritten in *foodInteraction* if a term with the semantic type *food* is found in its context.

We define a set of 44 contextual rewriting rules based on vocabulary used in questions from the training set and on the documentation of KBs, mainly of Drugbank [14]. After this processing, the identification of relevant elements can be performed more safely.

After the identification of relevant elements, we perform the question abstraction, for which we identify information re-

lated to the structure of query *(ie*, question topic, result form, predicate and arguments, scope of negation and conjunctions):

- The first entity with the semantic type that is provided by the RDF subjects and objects issued from the resources is considered to be the question topic. This information is used for the query construction. The question topic in Figure 3 is identified as *drug*;

Agregation operator:			
Question topic:	drug (drugbank/drugs)		
Predicates:	Frame		
	drugbank/drugs	state	STRING
Arguments:	Semantic type	Word	SPARQL type
	drugbank/state/Gas	gaseous	STRING/Gas

Figure 3: Identification of relevant elements in questions.

- For the definition of the result form, the question is scanned in order to identify words expressing negation (e.g. *no*), aggregation operation on the results (e.g. *number* for *count*, *mean* for *avg*, *higher* for *max)*, and specific result form such as boolean queries (*ASK).* As we can see, at this step the linguistic expressions are associated with the corresponding SPARQL operators. Information extracted at this step is recorded and used for the query construction. In example on Figure 3, no such information is found;

- Identification of Predicate and Argument: we use linguistic representation of the RDF schema i.e. frames which contain one predicate and at least two elements with associated semantic types. In that respect, the potential predicates, subjects and objects of frames are identified among the semantic entities and recorded: entries are the semantic types of the elements and refer to linguistic, semantic and SPARQL information associated with these elements. Subjects and objects are described with inflected and lemmatized forms of words or terms, the corresponding SPARQL types and indicators on their use as object or subject of a predicate. Concerning the predicates, only semantic types of their arguments are instantiated. Subjects and objects can be URI, RDF typed literals (numerical values or strings) and extracted terms (these are considered as elements of regular expressions). On Figure 1, the predicate *state* with the expected arguments *drugbank/drugs* and *Gas/String* is recognized.

- Scope of negation and conjunctions: Argument and predicate in the neighborhood of negation and conjunctions are identified. These elements are recorded as negated or coordinated.

Agregation operator:			
Question topic:	**?v0**		
Predicates:	Frame		
	?v0	state	**STRING/Gas**
Arguments:	Semantic type	Word	SPARQL type
	drugbank/state/Gas	gaseous	STRING/Gas

Figure 4: Construction of queries.

Query Construction. The objective of the query construction step is to connect previously identified elements and to build a semantic representation of the SPARQL graph pattern (introduced by the keyword *WHERE).* Figure 4 presents the architecture of the connection of elements and query construction. Thus, the predicate arguments are instantiated by URIs associated with the subjects, objects, variables, and numerical values or by strings. For each question, we perform several connections:

- The question topic is connected to the predicate(s). A given variable is associated with the question topic and with the predicate argument that matches the semantic type of the question topic. Note that at the end of this step, the question topic may remain non-associated to any predicate. In Figure 4, variable *?v0* represents the association between the question topic and the subject (with the expected type *drugbank/drugs*) of the predicate *state;*

- The predicate arguments are connected to subject and objects identified during the question abstraction: they concern elements referring to URI. Moreover, each predicate within the conjunction scope is duplicated and its arguments are also connected to the subject and objects if needed;

- The predicates are connected between them through their subjects and objects. The connection between two predicates is also represented by a variable;

- The predicates from different datasets are connected. We use the *sameAs* description to identify URIs referring to the same element. New variables are defined to connect two predicates;

- The remaining question topic is connected to arguments of the *sameAs* predicate;

- The arguments corresponding to the *string* type are connected with the extracted terms, that are considered as string expressions. They are connected through the string matching operator *REGEX.*

At this point, the predicate arguments which remain unassociated are replaced by new variables in order to avoid empty literals. Finally, the negation operator is processed: the predicates are marked as negated and the arguments within the negation scope are included in a new predicate *rdf:type* if required. At this step, each question is fully translated into a representation of the SPARQL query. Figure 5 illustrates the construction of the query.

```
SELECT DISTINCT ?v0
WHERE {
?v0     <http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugbank/state>
"Gas".
}
```

Figure 5: Construction of queries.

Query Generation. This is a syntactic step, during which the query is formed syntactically. The SPARQL query representation built during the query construction step is used to generate the SPARQL query string. The process is composed of two steps:

- Generation of the result form which takes into account the expected type of the result form *(ASK* or *SELECT*), the presence of aggregation operators and the variable associated to the question topic;

- Generation of the graph pattern. This step consists in generation of strings for representing each RDF triple and defining SPARQL filter if the predicates are negated. When aggregation operators are used, we also need to recursively generate sub-queries for computing the subsets of expressions, before their aggregation. In example from Figure 1, the predicate

state is replaced by the corresponding URI and its object is replaced by the string *gas*.

The SPARQL queries have been submitted to a SPARQL end-point. For these experiments, we use the SPARQL end-point provided by the QALD-4 challenge [23] and answers are collected for the evaluation.

Evaluation Metrics. The generated SPARQL queries are evaluated through their answers with three macro-measures [24]:

$$M_{\text{precision}} = \frac{\sum_{i=1}^{|q|} \frac{TP(q)}{TP(q)+FP(q)}}{|q|}$$

$$M_{\text{recall}} = \frac{\sum_{i=1}^{|q|} \frac{TP(q)}{TP(q)+FN(q)}}{|q|}$$

$$M_{\text{F-measure}} = \frac{2 \times M_{\text{precision}} \times M_{\text{recall}}}{M_{\text{precision}} + M_{\text{recall}}}$$

where *TP(q)* are the correct answers, *FP(q)* are the wrong answers and *FN(q)* are the missing answers for the question *q*. The use of macro-measures equally considers all the questions independently of the number of expected answers to the SPARQL queries. The comparison is done with the answers obtained through the manual querying of KBs.

Results

Global Results. Table 1 indicates the overall results obtained on the training and test sets. On the test set, the macro-F-measure is 0.78 with 0.81 precision and 0.76 recall while on the training set, the macro-F-measure is 0.86 with 0.84 precision and 0.87 recall. Each question is processed in less than 2 seconds on a standard computer (2.7GHz dual-core CPU and 4 Gb of memory). Most of the computing time is spent for the linguistic and semantic annotation of the questions.

Query set	Training (50 Q)	Test (27 Q)
Correct queries	39	20
Micro-precision	0.84	0.81
Micro-recall	0.87	0.76
Micro-F-measure	0.86	0.78

Table 1: Results obtained with the training and test sets.

Discussion

It must be noticed that questions from the test set may involve new operators and lexicon, not known from the test set. Nevertheless, our method always proposes syntactically correct SPARQL queries for all natural language questions from the training and test sets: 18 questions provide the exact expected answers, two questions return partial answers, other questions return no correct answers. On the training set, 39 SPARQL queries (out of the 50 questions) are semantically correct and provide the expected answers, 6 questions return partial answers, and 5 questions return no answers.

An analysis of erroneous or partial answers shows that most of the errors are due to:

(i) The encoding of the *sameAs* predicate in the resources. We observe that, although our method generates the correct SPARQL query, the SPARQL end-point does not return the expected answers. Besides, we observe that the correct answers can be obtained by the switching the arguments of the *sameAs* predicate in the queries. It appears thus that the instances of this predicate do not encode the expected reflexivity of this relation in the sources KBs while our method assumes that the *sameAs* predicate is reflexive by definition;

(ii) The management of ambiguities in questions. The errors due to the ambiguities are mainly related to:

- The annotation of semantic entities. For instance, in question *Which genes are associated with breast cancer?*, *breast cancer* is correctly annotated, while the reference data propose that it is to be associated with the semantic entity *Breast cancer-1*.
- The expected meaning of terms in the questions. Semantic entities that occur in some questions may refer to specific entities while in other questions they refer to general entities. For instance, the semantic entity *anemia* in *What are enzymes of drugs used for anemia?* refers to all types of anemia *Hypercholanemia*, *Hemolytic anemia*, *Aplastic anemia*, etc.), and not to the elements that contain the label *anemia.*

We defined and described some rules for managing the ambiguities, but they must be completed with additional rules. For instance, these two main problems can be solved by using regular expressions in SPARQL graph rather than URIs. However, we must test the influence of this modification on the whole set of queries. Other erroneous answers happen during the question abstraction step when the question topics are wrongly identified or when the contextual rewriting rules cannot be applied. Errors also occur during the query construction step: the method may abusively connect predicate arguments and semantic entities or, on contrary, it may not consider all the identified semantic entities. Further investigations have to be carried out to solve these limitations.

During the design of queries, we may also have difficulties to express some constraints in SPARQL. For instance, the question *Which approved drugs interact with calcium supplements?* requires to define a regular expression with the term *calcium supplement* while this term is only mentioned in conjunction with other supplement (*e.g. Do not take calcium, aluminum, magnesium or Iron supplements within 2 hours of taking this medication.*). We assume that solving this difficulty requires a more sophisticated NLP processing of the textual elements of the RDF triples: parsing of the RDF textual elements, named entity and term recognition, identification of discontinuous terms and term variants, etc.

When a sufficient quantity of questions and corresponding queries is available, some regular expressions used in the current work can be replaced by machine learning approaches in order to make a generalization of the named entity detection, or for the detection of semantic relations and operators.

Other limitations are related to the updating of the knowledge bases and the change of their structure. In the former case, it is only required to rebuild the semantic resources used for identifying the semantic entities. In the latter case, the entire frames must be regenerated. This is an ongoing research work. Moreover, the possibility to add new resources, such as Dailymed [25] or RxNorm [26], is a related problem. When managed, this functionality will allow processing new linked datasets with low adaptation cost.

Conclusion

We proposed a rule-based method to translate natural language questions into SPARQL queries. The method relies on linguistic and semantic annotations of questions with the NLP methods, semantic resources and the RDF triples description. We designed our approach on 50 biomedical questions proposed by the QALD-4 challenge, and tested it on 27 newly created questions. The method achieves good performance with 0.78 F-measure on the set of 27 questions.

Further work aims at addressing the limitations of our current method including the management of the term ambiguity, the question abstraction, and the query construction. Moreover, to avoid the manual definition of the dedicated resources required by our approach (frames, specific vocabulary and rewriting rules), we plan to investigate how to automatically build such dedicated resources from the RDF schemas of the Linked Data set. It will also facilitate the integration of other biomedical resources such as Dailymed or RxNorm, and the use of our method in text mining applications.

Acknowledgements

This work was partly funded through the project POMELO (PathOlogies, MEdicaments, aLimentatiOn) funded by the MESHS (Maison européenne des sciences de l'homme et de la société) under the framework Projets Emergents.

References

[1] Wishart, D.S., Knox, C., Guo, A.C., Shrivastava, S., Hassanali, M., Stothard, P., Chang, Z., Woolsey, J.: Drugbank: a comprehensive resource for in silico drug discovery and exploration. Nucleic Acids Research 34, D668–D672 (2006), database issue

[2] http://clinicaltrials.gov

[3] Kuhn, M., Campillos, M., Letunic, I., Jensen, L.J., Bork, P.: A side effect resource to capture phenotypic effects of drugs. Molecular Systems Biology 6(1) (2010), http://msb.embopress.org/content/6/1/343

[4] http://www.w3.org/wiki/HCLSIG/LODD

[5] Kaufmann, E., Bernstein, A.: How useful are natural language interfaces to the semantic web for casual end-users? In: Proceedings of the Forth European Semantic Web Conference (ESWC 2007). Innsbruck, Austria (June 2007)

[6] http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/index.php?x=task2&q=4

[7] Pradel, C., Haemmerlé, O., Hernandez, N.: Des patrons modulaires de requêtes SPARQL dans le système SWIP. In: Journées Francophones d'Ingénierie des Connaissances (IC). pp. 412--428 (juin 2012)

[8] Lehmann, J., Bühmann, L.: Autosparql: Let users query your knowledge base. In: Proceedings of the 8th extended semantic web conference on The semantic web: research and applications. vol. Part I, pp. 63--79 (2011)

[9] Unger, C., Bühmann, L., Lehmann, J., Ngomo, A.C.N., Gerber, D., Cimiano, P.: Template-based question answering over RDF data. In: WWW. pp. 639—648 (2012)

[10] Damljanovic, D., Agatonovic, M., Cunningham, H.: Natural language interfaces to ontologies: Combining syntactic analysis and ontology-based lookup through the user interaction. In: Proceedings of the 7th International Conference on The Semantic Web: Research and Applications - Volume Part I. pp. 106--120. ESWC'10, Springer-Verlag, Berlin, Heidelberg (2010)

[11] Kuchmann-Beauger, N., Aufaure, M.A.: Natural language interfaces for datawarehouses. In: 8èmes journées francophones sur les Entrepôts de Données et l'Analyse en ligne (EDA 2012), Bordeaux. RNTI, vol. B-8, pp. 83--92. Hermann, Paris (Juin 2012)

[12] Abacha, A.B., Zweigenbaum, P.: Medical question answering: Translating medical questions into sparql queries. In: ACM SIGHIT International Health Informatics Symposium (IHI 2012) (2012)

[13] Biomedical question answering over interlinked data, http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/index.php?x=task2&q=4

[14] http://www.drugbank.ca/documentation

[15] http://diseasome.eu

[16] Janjić, V., Pržulj, N.: The core diseasome. Mol Biosyst 8(10), 2614-2625 (Aug 2012), http://dx.doi.org/10.1039/c2mb25230a

[17] http://sideeffects.embl.de

[18] Kuhn, M., Campillos, M., Letunic, I., Jensen, L.J., Bork, P.: A side effect resource to capture phenotypic effects of drugs. Molecular Systems Biology 6(1) (2010), http://msb.embopress.org/content/6/1/343

[19] Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In: Jones, D., Somers, H. (eds.) New Methods in Language Processing Studies in Computational Linguistics (1997)

[20] http://search.cpan.org/~thhamon/Alvis-TermTagger/

[21] Aubin, S., Hamon, T.: Improving term extraction with terminological resources. In: Salakoski, T., Ginter, F., Pyysalo, S., Pahikkala, T. (eds.) Advances in Natural Language Processing (5th International Conference on NLP, FinTAL 2006). pp. 380--387. No. 4139 in LNAI, Springer (August 2006)

[22] http://search.cpan.org/~thhamon/Lingua-YaTeA/

[23] http://vtentacle.techfak.uni-bielefeld.de:443/sparql (Last accessed November 2014)

[24] Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1--47 (2002)

[25] http://dailymed.nlm.nih.gov/

[26] National Library of Medicine. RxNorm, a standardized nomenclature for clinical drugs. 2009. Available at www.nlm.nih.gov/research/umls/rxnorm/docs/index.html. Bethesda, Maryland, USA

Address for correspondence

Thierry Hamon

LIMSI-CNRS
Rue John von Neumann
Campus Universitaire d'Orsay
Bât 508
91405 ORSAY, Cedex, France
thierry.hamon@limsi.fr

+33 (0) 1 69 85 80 39