Natural Language Question Analysis for Querying Biomedical Linked Data

Thierry Hamon[1,2], Natalia Grabar[3], and Fleur Mougin[4]

[1] LIMSI-CNRS, Orsay, France
thierry.hamon@limsi.fr,
[2] Université Paris 13, Sorbonne Paris Cité, France
[3] STL UMR8163 CNRS, Université Lille 3, France
natalia.grabar@univ-lille3.fr,
[4] Université Bordeaux, ISPED, Centre INSERM U897, ERIAS, France
firstname.lastname@isped.u-bordeaux2.fr

Abstract. Biomedical knowledge is disseminated in knowledge bases which become increasingly available on the Web. While using biomedical Linked Data is crucial, life-science researchers may have difficulties using SPARQL language. Interfaces based on Natural Language question-answering are recognised to be suitable for querying knowledge bases. In this paper, 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. We designed a four-step method which linguistically and semantically annotate the question, performs an abstraction of the question, then builds a representation of the SPARQL query and finally generates the query. The method is designed on 50 questions over 3 biomedical knowledge bases in the task 2 of the QALD-4 challenge framework and evaluated on 27 new questions. It achieves good performance with 0.78 F-measure on the test set. The method for translating questions into SPARQL queries is implemented as a Perl module and is available at http://search.cpan.org/~thhamon/RDF-NLP-SPARQLQuery/.

Keywords: Natural Language Processing, SPARQL, biomedical domain, semantic resources, frames

1 Introduction

The knowledge bases (KB) recording biomedical knowledge are becoming increasingly available on the Web. Such life-science bases usually focus on a specific type of biomedical information: chemical, pharmacological and target information on drugs in Drugbank [15], clinical studies in ClinicalTrials.gov[5], drugs and their side effects in Sider [8], etc.

Nowadays, the connections between these knowledge bases are crucial for obtaining a more global and comprehensive view on the links between different biomedical components. They are also required for inducing and producing

[5] http://clinicaltrials.gov/

new knowledge from the already available data. Particularly, the creation of fine-grained links between the existing knowledge bases related to drugs is a great challenge that is being addressed by the project Linked Open Drug Data (LODD) for instance[6]. The knowledge recorded in the KB and dataset interlinks are represented as RDF triples, on the basis of which the linked data can then be queried through a SPARQL end-point. However, typical users of this knowledge, such as physicians, life-science researchers or even patients, cannot manage the syntactic and semantic requirements of the SPARQL language neither can they manage the structure of various knowledge bases. This situation impedes the efficient use of knowledge bases and the retrieval of useful information. Therefore, it is important to design friendly interfaces that mediate the technical and semantic complexity of the task and provide simple approaches for querying the knowledge bases. The main challenge is then to design the optimal methodology for an easy and reproducible rewriting of natural language questions in SPARQL queries.

We present in this paper the method to translate natural language questions into SPARQL queries over biomedical Linked Data. The method is based on the use of Natural Language Processing (NLP) tools and resources to enrich question with linguistic and semantic information. Questions are then translated in SPARQL with a rule based approach. We design our approach on the 50 questions proposed by the task 2, *Biomedical question answering over interlinked data*,[7] of the QALD-4 challenge, and evaluate it on 27 newly defined questions. Sample of the new test set is given at Sect. 5.1. We work with three KBs (Drugbank, Diseasome, and Sider described in Sect. 4).

The paper is structured as follows. Section 2 presents the related work. We describe the proposed method in Sect. 3 and, then, the semantic resources available and developed for enriching the questions in Sect. 4. The evaluation of the method is presented in Sect. 5.

2 Related Work

Querying Linked Data requires the definition of end user interface which hides the underlying structure of the knowledge bases as well as the SPARQL syntax. While [9] identify three ways for querying Linked Data (Knowledge-Based Specific Interface, Graphical Query Builder and Question-Answering System), it has been recognised that Natural Language interfaces are the most suitable. Thus, [6] demonstrate that for querying the knowledge bases and the Semantic Web data, the use of full and standard sentences is preferred to the use of keywords, menus or graphs.

Also, [9] propose a Question-Answering system (AutoSPARQL) based on active supervised machine learning and independent of the knowledge bases: SPARQL query model is learnt from Natural Language questions. The authors report that 50 questions are successfully processed with the system.

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

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

Most of the approaches usually rely on patterns or templates for translating the natural language questions into SPARQL queries. Thus, [11] design modular patterns to parse questions. Their main purpose is to model the questions for the SPARQL query generation where first keywords are detected and then relations between them are identified. The method is tested on 160 movie related questions. The current model requires 4 query patterns instead of 12 patterns proposed in the previous work.

[14] observe that Question-Answering systems offer a good compromise between expressivity and intuitiveness, and also propose a template-based system relying on the NLP tools and semantic resources to process natural language questions. The application of the system on 50 questions on DBpedia proposed by the QALD-2 challenge gives competitive results with 0.62 average F-measure obtained with 39 questions (average recall is 0.63 and average precision is 0.61), but suffer of a low coverage (the 11 other questions were not covered by the templates).

Up to now the design of friendly user interface is mainly addressed for general knowledge bases [4], [7]. However, as domain-specific Linked Data are more and more available, the studies on these data appear. Thus, we can mention a work which aim is to translate medical questions issued from a journal into SPARQL queries [1]. The method combines the SVM machine learning based approach to extract the characteristics of the questions (named entities, relations) with patterns to generate the SPARQL queries. The evaluation is carried out on 100 questions and the corresponding queries on clinical documents. The method achieves a 0.62 precision when querying the documents. Recently, the Question Answering over Linked Data (QALD-4) challenge proposes the task[8] dedicated to the retrieval of precise biomedical information in linked knowledge bases with questions in natural language.

Our approach is close to the one proposed by [14] as we also use the NLP tools to linguistically enrich the questions. The main difference is that we use information issued from the Linked Data resources to semantically annotate the questions and to define the frames (i.e. linguistic representation of the RDF schema) in order to model and build the SPARQL.

3 Question translation into SPARQL Query

To translate natural language questions into SPARQL queries, we design a four-step rule-based method relying on NLP methods, semantic resources and the RDF triple description (see Fig. 1). Natural language questions are enriched with linguistic and semantic information (Sect. 3.1). This information is used to abstract the questions and identify the relevant information (Sect. 3.3), to build the corresponding SPARQL query representation (Sect. 3.3) and to generate the SPARQL query (Sect. 3.4). The same process is used for translating the questions from the training set as well those issued from the test set. The method for

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

translating questions into SPARQL queries is implemented as a Perl module and is available at http://search.cpan.org/~thhamon/RDF-NLP-SPARQLQuery/.

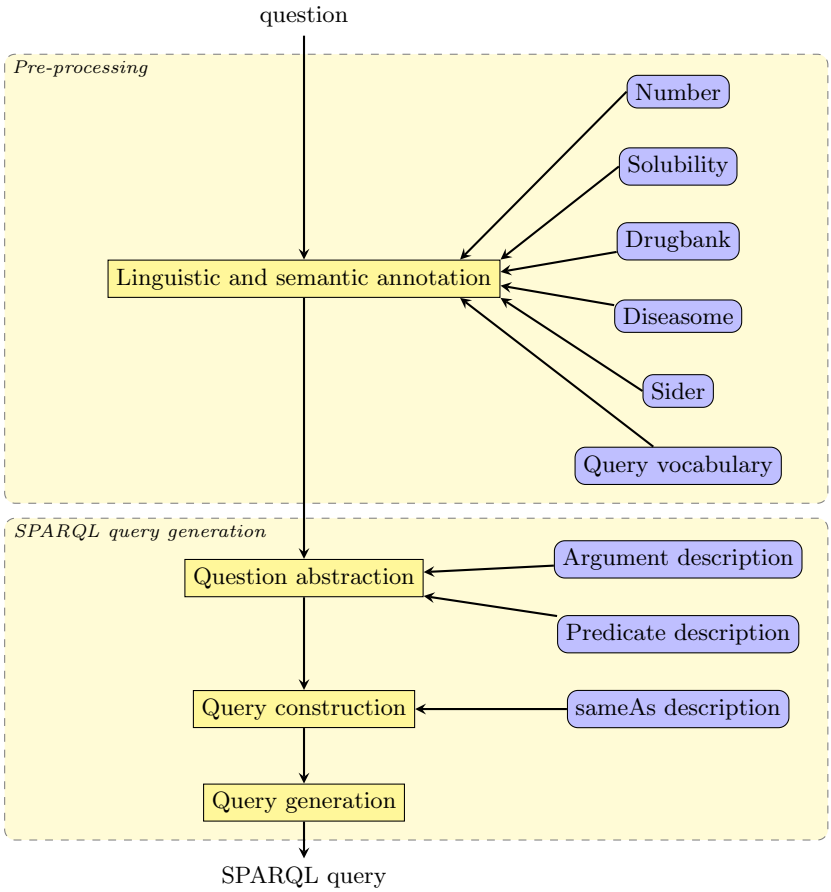


Fig. 1. The global architecture of the system (the processing steps are in yellow, the resources in blue). Yellow boxes represent the processing steps which are further detailed in figures 2 to 6. Blue boxes mean the resource used for the processing of the questions and queries.

3.1 Linguistic and Semantic Annotation of the Questions

The annotation step aims at associating linguistic and semantic description to the words and terms from the question (see Fig. 2). First, numerical values (such as numbers and solubility values) are identified with a named entity recogniser.

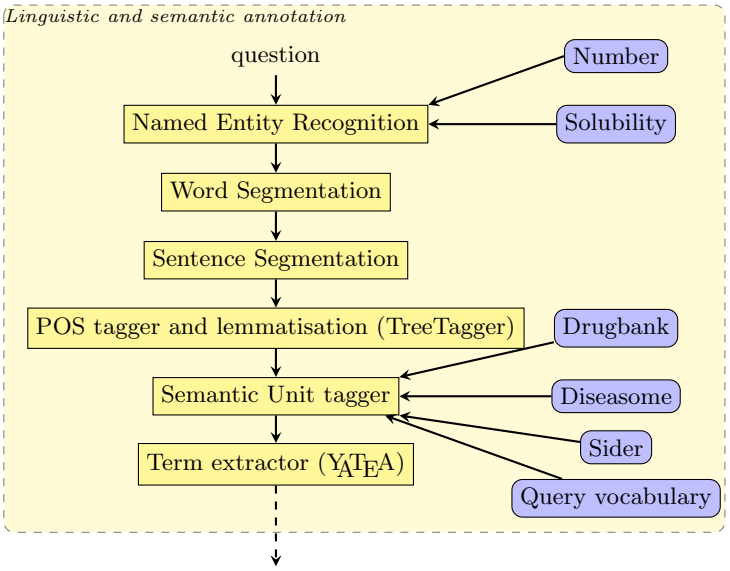


Fig. 2. Linguistic and semantic annotation process. Yellow boxes represent the steps for linguistic and semantic analysis of questions. Blue boxes indicate the resources used for the semantic annotation.

Then, the parsing of the questions consists in the word segmentation, the part-of-speech tagging and the lemmatization of the words with TreeTagger [12]. Semantic entities, i.e. terms with associated semantic types representing their meaning, are identified with the TermTagger Perl module[9]. This term recognition relies on the semantic resources (see Sect. 4) to recognize semantic entities such as disease names, side effects, etc. However, because the semantic resources often suffer from low coverage [3, 10], we also use the term extractor YaTeA[10] [2] to improve the coverage of our method. YaTeA performs shallow parsing of the POS-tagged and lemmatized text by chunking it according to syntactic frontiers (pronouns, conjugated verbs, typographic marks, etc.) in order to identify noun phrases. Then, parsing patterns are recursively applied and provide parsed terminological entities, usually noun phrases relevant for the targeted domain. These parsing patterns have been manually defined during previous work. They take into account the morpho-syntactic variation and reflect basic syntactic dependency in terminological entities. Each term is represented as syntactic tree, and sub-terms are also considered as terms in the current configuration. No semantic types are associated to the terms extracted by YaTeA. Figure 3 illustrates the linguistic and semantic annotation of a question.

[9] http://search.cpan.org/~thhamon/Alvis-TermTagger/
[10] http://search.cpan.org/~thhamon/Lingua-YaTeA/

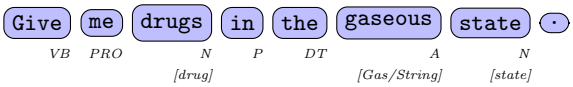


Fig. 3. Example of question pre-processing (question#22 of the QALD-4 test set). The blue boxes represent the word and semantic entites. The subscript texts are the Part-of-Speech tags and the bracketed subscript texts are the semantic types associated to the semantic entities.

3.2 Question Abstraction

The question abstraction step aims at identifying the relevant elements within the questions and at building the representation of these elements (see Fig. 4). It relies on the linguistic and semantic annotations associated to the question words in the previous step.

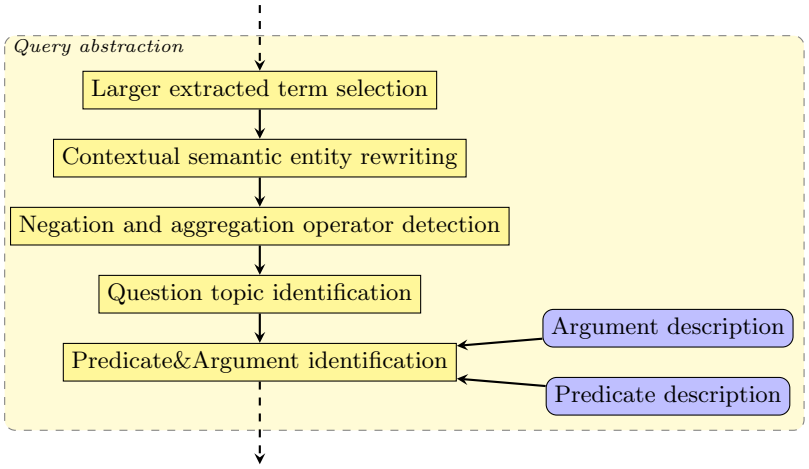


Fig. 4. Question abstraction process. Yellow boxes represent the abstraction steps of questions. Blue boxes indicate the resources used.

Before the identification of the relevant elements, the annotations are post-processed in order to disambiguate the semantic annotations. Indeed, the annotated semantic entities may receive conflicting or concurrent semantic types, while the post-processing permits to select those entities and semantic types that may be useful for the next steps. We keep larger terms which do not include other semantic entities. Also, we manually defined rewriting rules on the training set in order to modify or delete the semantic type associated with an entity according to the context. Other rules may also modify or delete the entity according to the context. For instance, the semantic entity *interaction* has to be

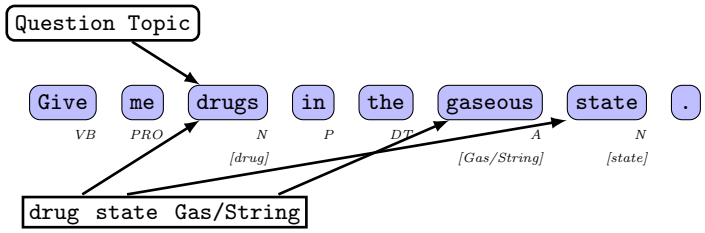
rewritten in `interactionDrug1` if its context contains mention of drug, but it has to be rewritten in `foodInteraction` if its context contains a term with the semantic type *food*. On the whole, we defined 44 contextual rewriting rules based on the vocabulary used in the questions and on the documentation of knowledge bases, mainly the one from Drugbank[11]. Besides the rewriting rules, additional disambiguation of the annotations is also performed during the *SPARQL query construction* step when the arguments of the predicate or the question topic are connected as they share the same semantic.

For performing the abstraction of questions, we identify information related to the query structure:

1. Definition of the Result form: the question is scanned for identifying words expressing the negation, e.g. *no*, and its scope, the aggregation operation on the results, e.g. *number* for `count`, *mean* for `avg` or *higher* for `max`, and specific result form such as boolean queries (`ASK`). The information concerning the presence of the negation or conjunction marks and aggregation operators but also the requirement of specific result form is recorded to be used at the end of the *query construction* step or during the *query generation* step. In the example on figure 3, no such information is found.
2. Identification of the Question topic: the first semantic entity with a given expected semantic type is considered as the question topic (this assumption is alsways verifed in the training set but also the test set). The expected semantic types are those provided by the RDF subjects and objects issued from the resources. This information will be used during the *query construction* step. The question topic of the example is identified as **drug**.
3. Identification of Predicate and Argument: we consider the 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 in the table (entries are the semantic type of the elements and refer to linguistic, semantic and SPARQL information associated with the elements). The subjects and objects are fully described in the table with the inflected and lemmatized form of the word or term, the corresponding SPARQL type and the indicators of the use as object or subject of a predicate. Concerning the predicates, only the semantic types of their arguments are instantiated. The subjects and objects can be URI, RDF typed literals (numerical values or strings) and extracted terms (these are considered as elements of regular expressions). In the example from figure 3, the predicate **state** with the expected arguments **drugbank/drugs** and **Gas/String** is recognised.
4. Scope of the negation and conjunction: Argument and predicate in the neighbourhood of negation of conjunction are identified. These elements are recorded as negated.

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

Figure 5 presents a graphical representation and the abstraction of the question 22 of the QALD-4 test set.



Aggregation operator:	
Question topic:	drug (`drugbank/drugs`)
Predicates:	Frame
	`drugbank/drugs` state `Gas/String`
Arguments:	Semantic type Word SPARQL type
	`Gas/String` gaseous STRING[Gas]

Fig. 5. Example of question abstraction (question#22 of the QALD-4 test set)

3.3 Query Construction

The objective of the *query construction* step is to connect previously identified elements and to build a representation of the SPARQL graph pattern (introduced by the keyword `WHERE`). Figure 6 presents the architecture of the query construction. Thus, the predicate arguments are instantiated by either the URI associated with the subjects, objects, variables, and numerical values or the strings. For each question, we perform several connections:

1. The question topic is connected to the predicate(s). A variable is associated with the question topic and the predicate arguments that matched the semantic type of the question topic. Note that at the end of this stage, the question topic may remain non-associated with any predicate. In the example from figure 3, the variable `?v0` represents the association between the question topic and the subject (with the expected type **drugbank/drugs**) of the predicate **state**.
2. The predicate arguments are connected to the subjects and objects identified during the question abstraction: they concern elements referring to URI. Moreover, each predicate in the conjunction scope is duplicated and arguments are also connected to the subject and objects if needed;
3. The predicates are connected between them through their subjects and objects. The connection between two predicates is then represented by a variable;

4. The predicates from different datasets are connected. We use the `sameAs` description to identify URI referring to the same element. New variables are defined to connect two predicates;
5. The remaining question topic is connected to arguments of the `sameAs` predicate;
6. The arguments are connected to the `string` type to extracted terms annotated in the question. We assume these arguments will be related to the string matching operator `REGEX`. Thus, the terms are considered as string expressions.

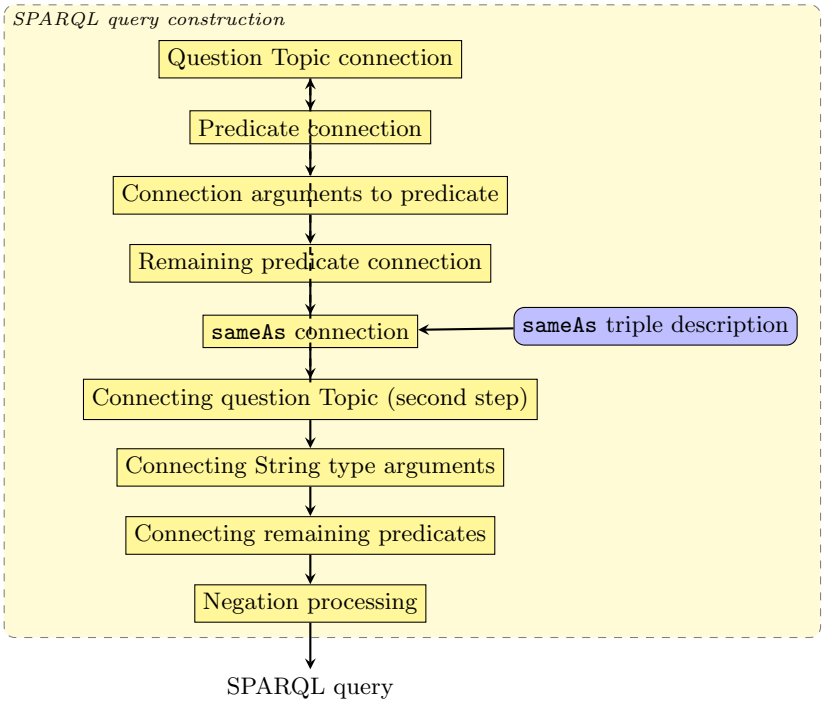
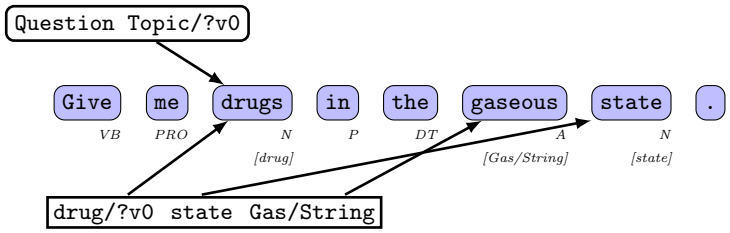


Fig. 6. Query construction process. Yellow boxes represent the construction steps of queries. Blue boxes indicate the resource used.

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 stage, each question is fully translated into a representation of the SPARQL query. Figure 7 illustrates the construction of the query corresponding to the question 22 of the QALD-4 test set.



Aggregation operator:			
Question topic:	?v0		
Predicates:	Frame		
	?v0 state `Gas/String`		
Arguments:	Semantic type	Word	SPARQL type
	`Gas/String`	gaseous	STRING[Gas]

Fig. 7. Example of query construction (question#22 of the QALD-4 test set)

3.4 Query Generation

The SPARQL query representation built during the *query construction* step is used to generate the SPARQL query string. It is composed of two parts:

1. The 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;
2. The generation of the graph pattern. This part consists in the generation of the strings for representing each RDF triple and the filtering if the predicates are negated. But when aggregation operators are used, we also need to recursively generate sub-queries for computing the subsets of expressions, before their aggregation. In the example from figure 3, 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[12] and answers are collected for the evaluation. Figure 8 presents the generated query which corresponds to the question 22 of the QALD-4 test set.

4 Definition of the Semantic Resources

The method described above relies on the existing biomedical resources that provide information on the semantic entities (Sect. 4.1), but also on additional resources specifically collected and built to support the method (Sect. 4.2).

[12] for our experiments, we use the SPARQL end-point provided by the QALD-4 challenge http://vtentacle.techfak.uni-bielefeld.de:443/sparql

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

Fig. 8. Example of query generation (question#22 of the QALD-4 test set)

4.1 Domain-specific Resources

Regarding the QALD-4 question, we use three biomedical resources:

- Drugbank[13] is dedicated to drugs [15]. It merges chemical, pharmacological and pharmaceutical information from other available knowledge bases. We exploited the documentation[14] of this resource to define 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 [5]. It provides a single framework with all known phenotypes and disease gene associations, indicating the common genetic origin of many diseases. We exploited the RDF triples and the documentation of the resource to define the rewriting rules.
- Sider[16] is dedicated to adverse drug effects [8]. It contains information on marketed medicines and their recorded adverse drug reactions. The information is extracted from public documents and package inserts. The available information includes side effect frequency, drug and side effect classifications as well as links to other information, for example drug-target relations.

The content of each resource is provided in a specific format: RDF triples *subject predicate object*. In that respect, we also exploit their RDF schema to define frames (see below).

4.2 Additional Resources for the Question Annotation

On the basis of the RDF triples, frames are built from the RDF schema where 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:

- As indicated, subject, object and predicate 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 operators, negation and types of questions.

[13] http://www.drugbank.ca
[14] http://www.drugbank.ca/documentation
[15] http://diseasome.eu
[16] http://sideeffects.embl.de

– RDF literals, issued from named entity recognizer or term extractor, complete the resources. The RDF literals are detected with specifically designed automata that may rely on the source knowledge base documentation.

These entities are associated with the expected semantic type, which allows creating the queries and rewriting the RDF triples in the SPARQL queries. In that respect, we can consider IRI, strings, common datatype or regular expressions when literals are expected.

Most of the entities are considered and processed through their semantic types, although some ambiguous entities (e.g. interaction or class) are considered atomically. For these, the rewriting rules will be applied contextually to generate the semantic entities corresponding to the frames (see Sect. 3.2). When using the queries, the semantic types are variables and are used for connecting the edges of queries.

5 Experiment and Results

5.1 Training and Test question set

As the complexity of the QALD-4 training and test was unbalanced (e.g. the QALD-4 training set does not propose questions requiring the use of aggregation operators), we consider the 50 questions of the QALD-4 training and test sets for training. Besides, similarly to the usual layout of challenge data, we design 27 new questions for the evaluation. The questions of the new test are similar to the QALD-4 questions but may differ according to the semantic entities or the involved predicates. Our method processes this new test set without considering additional adaptation. Figure 9 presents a sample of questions from the new test set.

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?

Fig. 9. Example of natural language questions from the new test set

5.2 Evaluation Metrics

The generated SPARQL queries are evaluated through their answers with following macro-measures [13]:

$$M\text{-precision} = \frac{\sum_{i=1}^{|q|} \frac{TP(q)}{TP(q)+FP(q)}}{|q|} \qquad M\text{-recall} = \frac{\sum_{i=1}^{|q|} \frac{TP(q)}{TP(q)+FN(q)}}{|q|}$$

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

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.

5.3 Global Results

Table 1 present the overall results 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 in 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.

Table 1. Results on the training set and the test set

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

The method always proposes syntactically correct SPARQL queries for each natural language question: 18 questions provide the exact expected answers, two questions return partial answers, other questions return no correct answers. On the training, 39 SPARQL queries (out of the 50 questions) are semantically correct and provide the expected answers. We can also observe that 6 questions return partial answers, and 5 questions return no answers.

Error Analysis The analysis of the erroneous or partial answers shows that most of the errors are due to (i) the encoding of the `sameAs` predicate in the resources and (ii) the management of ambiguities in the questions.

Regarding the first type of errors, we observe that, although our method generate the correct SPARQL query, the SPARQL end-point does not return the expected answers. Besides, we observe that by switching the arguments of the `sameAs` predicate in the queries, the expected answers are returned. In that respect, we consider the instances of this predicate do not encode the reflexivity of this relation in the resources while our method assumes that the `sameAs` predicate is reflexive by definition.

As for the errors due to the ambiguity, they mainly concern:

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

These two main problems could be solved by using regular expressions in SPARQL graph rather than URIs. However, we must test the influence of this modification on all the queries.

Other erroneous answers happen during the question abstraction where the question topic is wrongly identified and the contextual rewriting rule is not applied. Errors also occur during the query construction: the method abusively connects predicate arguments and semantic entities, or on contrary, does not consider all the identified semantic entities. Further investigations have to be carried out to solve these limitations.

Besides, during the design of queries, we had difficulties to express constraints in SPARQL. Thus, the question *Which approved drugs interact with calcium supplements?* requires to define a regular expression with the term *calcium supplement* while the 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.).

Other limitations concern 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 frames must be regenerated. This is an ongoing research work. Moreover, the addition of new resources as Dailymed[17] also addresses these two problems.

6 Conclusion

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

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

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 these dedicated resources from the RDF schema of 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.

Acknowledgments

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 Émergents*.

References

1. Abacha, A.B., Zweigenbaum, P.: Medical question answering: Translating medical questions into sparql queries. In: ACM SIGHIT International Health Informatics Symposium (IHI 2012) (2012)
2. 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)
3. Bodenreider, O., Rindflesch, T.C., Burgun, A.: Unsupervised, corpus-based method for extending a biomedical terminology. In: Workshop on Natural Language Processing in the Biomedical Domain (ACL2002). pp. 53–60 (2002)
4. 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)
5. Janji, V., Prulj, N.: The core diseasome. Mol Biosyst 8(10), 2614–2625 (Aug 2012), http://dx.doi.org/10.1039/c2mb25230a
6. 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)
7. Kuchmann-Beauger, N., Aufaure, M.A.: Natural language interfaces for datawarehouses. In: 8mes journes francophones sur les Entrepts de Donnes et l'Analyse en ligne (EDA 2012), Bordeaux. RNTI, vol. B-8, pp. 83–92. Hermann, Paris (Juin 2012)
8. 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
9. Lehmann, J., Bhmann, 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)
10. McCray, A.T., Browne, A.C., Bodenreider, O.: The lexical properties of the gene ontology (GO). In: Proceedings of the AMIA 2002 Annual Symposium. pp. 504–508 (2002)

11. Pradel, C., Haemmerl, O., Hernandez, N.: Des patrons modulaires de requtes SPARQL dans le systme SWIP. In: Journes Francophones d'Ingnierie des Connaissances (IC). pp. 412–428 (juin 2012)
12. 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)
13. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
14. 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)
15. 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, D668D672 (2006), database issue