FREyA: An Interactive Way of Querying Linked Data Using Natural Language

Danica Damljanovic[1], Milan Agatonovic[2], and Hamish Cunningham[1]

[1] Department of Computer Science, University of Sheffield
Regent Court, 211 Portobello Street, Sheffield, UK
{d.damljanovic,h.cunningham}@dcs.shef.ac.uk
[2] Fizzback, London, United Kingdom
magatonovic@fizzback.com

Abstract. Natural Language Interfaces are increasingly relevant for information systems fronting rich structured data stores such as RDF and OWL repositories, mainly because of the conception of them being intuitive for human. In the previous work, we developed FREyA, an interactive Natural Language Interface for querying ontologies. It uses syntactic parsing in combination with the ontology-based lookup in order to interpret the question, and involves the user if necessary. The user's choices are used for training the system in order to improve its performance over time. In this paper, we discuss the suitability of FREyA to query the Linked Open Data. We report its performance in terms of precision and recall using the MusicBrainz and DBpedia datasets.

Keywords: Natural language interfaces, ontologies, question-answering, learning, clarification dialogs.

1 Introduction

With the rapid growth of the Linked Open Data (LOD) cloud[1] the effective exploitation becomes an issue largely because of the complexity and syntactic unfamiliarity of the underlying triple models and the query languages built on top of them. Natural Language Interface (NLI) systems become increasingly relevant for information systems fronting rich structured data stores such as RDF and OWL repositories, mainly because of the conception of them being intuitive for human.

According to [8], a major challenge when building NLIs is to provide the information the system needs to bridge the gap between the way *the user* thinks about the domain of discourse and the way *the domain knowledge is structured* for computer processing. This implies that in the context of NLIs to ontologies, it is very important to consider the ontology structure and content. Two ontologies describing identical domains (e.g., music) can use different modelling conventions. For example, while one ontology can use a datatype property *artistName* of class *Artist*, the other one might use instances of a special class to model

[1] http://linkeddata.org

R. García-Castro et al. (Eds.): ESWC 2011 Workshops, LNCS 7117, pp. 125–138, 2012.
© Springer-Verlag Berlin Heidelberg 2012

the artist's name[2]. A *portable* NLI system would have to support both types of conventions without sacrificing performance. *Portable* NLIs are those that can be adapted easily to new domains (or new ontologies covering the same domains). Constructing such systems poses a number of technical and theoretical problems because many of the techniques developed for specialised systems preclude automatic adaptation to new domains [8].

Ontologies can be constructed to include sufficient lexical information to support a domain-independent query analysis engine. However, due to different processes used to generate ontologies, the lexicon might be of varying quality. In addition, some words might have different meanings in two different domains or context. For example, *How big* might refer to *height*, but also to *length*, *area*, or *population* – depending on the question context, but also on the ontology structure. This kind of adjustments – or mappings from words or phrases to ontology concepts/relations, is usually performed during the *customisation* of NLIs.

Many NLIs for querying ontologies have been developed in recent years. Challenges related to Natural Language understanding such as *ambiguity* and *expressiveness* are balanced by constraining the supported language, e.g. by using a Controlled Natural Language, such as in AquaLog [14], or ORAKEL [1]. While NLI systems with a good performance require customisation (such as in the case of ORAKEL), several systems have been developed for which the customisation is not mandatory (e.g., AquaLog, PANTO [17], Querix [10], NLP-Reduce [10], QuestIO [5]). However, as reported in [14] the customisation usually improves recall. On the other hand, the complexity of supported questions differs from one system to another. While systems such as NLP-Reduce or QuestIO process queries without deep grammar analysis, the other systems (such as ORAKEL) support compositional semantic constructions such as quantification and negation.

With regards to *portability*, most of these systems are tested in the *closed-domain* scenario with ontologies which cover different, but narrow domains, with the exception of PowerAqua [15], the system that evolved from AquaLog aiming to serve as a Question-Answering system for the Semantic Web. PowerAqua was evaluated in the *open-domain* scenario [12] (e.g. through querying the ontologies indexed by Watson [6]). Portability of the majority of other NLIs to ontologies is tested by demonstrating that all that is required to port the system is the ontology URI – the system automatically generates the *domain lexicon* by reading and processing ontology lexicalisations.

With the availability of Linked Open Data, *portability* gained a new dimension bringing up the open-domain scenario where the context is multiple domains/ontologies on the contrary to the previously considered closed-domain. Having more than one ontology describing exactly the same domain, or hundreds of domains in one huge dataset requires support for *heterogeneity, redundancy* and *incompleteness* which comes with this multi-billion dataset. In other words, the system now needs to deal not only with how to map certain terms to the ontology concepts but it also needs to disambiguate and decide

[2] See for example how class *Alias* is used in the Proton System Module ontology: `http://proton.semanticweb.org/`

which ontology should provide the best answer (should *Where* be mapped to *http://purl.org/dc/terms/Location*, *http://dbpedia.org/ontology/locationCity* or any other). On the other hand, availability of such enormous knowledge base gives the possibility to merge experience which has been collected for decades by researching open-domain Question-Answering systems, NLIs to databases, and dialog systems in order to successfully accomplish what has been a great challenge for such a long time: answering questions automatically using the distributed sources on the Web. This has not been possible with databases as they are distributed and not interoperable, while Question-Answering systems use methods from Information Retrieval to locate documents in which the answer may appear. Information Retrieval methods although scale well, do not often capture enough semantics - relevant documents could be easily disregarded if the answer is hidden in a form which is not in-line with the expected patterns.

In this paper, we discuss requirements and suitability for querying the Linked Open Data using the system called FREyA. FREyA is named after **F**eedback, **R**efinement and **E**xtended Vocabular**Y** **A**ggregation [4], as it aims to investigate whether *user interaction* coupled with deeper syntactic analysis and usability methods such as *feedback* and *clarification dialogs* can be used in combination to improve the performance of NLIs to ontologies. FREyA has previously shown a good performance (recall and precision reaching 92.4% [4]) on the Mooney GeoQuery dataset which is extensively used for the evaluation of NLIs in recent years. We report the performance of FREyA using the MusicBrainz and DBpedia datasets provided within the QALD-1 challenge[3] and discuss how we begin to address the problem of querying the linked data in the open-domain scenario.

2 FREyA

FREyA is an interactive Natural Language Interface for querying ontologies which combines usability enhancement methods such as *feedback* and *clarification dialogs* in an attempt to: 1) **improve recall** by enriching the domain lexicon from the user's vocabulary (see [2]) 2) **improve precision** by resolving *ambiguities* more effectively through the dialog. The suggestions shown to the user are found through ontology reasoning and are initially ranked using the combination of string similarity and synonym detection (using WordNet[7]). The system then learns from the user's selections, and improves its performance over time. In what follows we give a brief overview of FREyA, followed by the requirements for using the system with different datasets and challenges raised by using it with the linked data.

Figure 1 shows the workflow starting with a Natural Language (NL) question (or its fragment), and ending when the answer is found. **The syntactic parsing and analysis** generates a parse tree using Stanford Parser [11] and then uses several heuristic rules in order to identify *Potential Ontology Concepts (POCs)*. POCs refer to question terms/phrases which can but not necessarily have to be linked to Ontology Concepts (OCs). POCs are chosen based on the

[3] `http://www.sc.cit-ec.uni-bielefeld.de/qald-1`

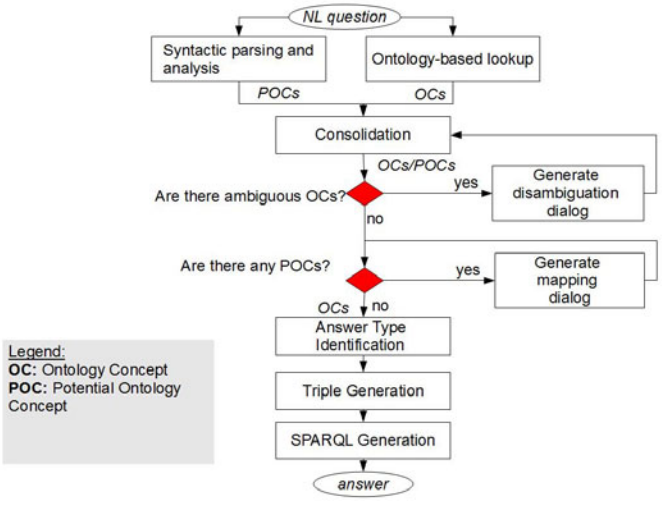

Fig. 1. FREyA Workflow

analysis of the syntactic parse tree, however this analysis does not require strict adherence to syntax and works on ill-formed questions and question fragments as well as on the grammatically correct ones. For example, nouns, verbs, or WH-phrases such as *Where*, *Who*, *When*, *How many* are expected to be found by our **POC identification algorithm**. This algorithm is based on the identification of prepreterminals and preterminals in the parsed tree, and also on their part-of-speech tags (see [4]).

The ontology-based lookup links question terms to logical forms in the ontology which we call Ontology Concepts (OCs) without considering any context or grammar used in the question (apart from morphological analysis, see [5]). Ontology Concepts refer to *instances/individuals*, *classes*, *properties*, or *datatype property values* such as string literals. By default, the system assumes that *rdfs:label* property is used to name the specific Ontology Concept. However, for ontologies which use different naming conventions (such as using *dc:title* inside the MusicBrainz dataset), it is possible to predefine which properties are used for names. This will enable the system to make the distinction between making a *datatype property value element* and an *instance element*. This distinction is important as different *elements* are used differently during the *Triple Generation* and *SPARQL generation* steps.

The consolidation algorithm aims at mapping existing POCs to OCs automatically. If it fails, the user will be engaged in the dialog. In the query *Give me all former members of the Berliner Philharmoniker.*, the **POC identification algorithm** will find that *the Berliner Philharmoniker* is a POC, while the **ontology-based lookup** will find that *Berliner Philharmoniker* is an OC, referring to an instance of *mm:Artist*. As the only difference in the POC and the OC text is a determiner (*the*), the consolidation algorithm will resolve this POC

automatically by removing it, and by verifying that this noun phrase refers to the OC with *dc:title Berliner Philharmoniker*.

When the system fails to automatically generate the answer (or when it is configured to work in the *forceDialog* mode, see Section 2.1) it will prompt the user with a dialog. There are two kinds of dialogs in FREyA. The **disambiguation dialog** involves the user to resolve identified ambiguities. The **mapping dialog** involves the user to map a POC to the one of the suggested OCs. While the two types of dialogs look identical from the user's point of view, there are differences which we will highlight here. Firstly, we give a higher priority to the disambiguation dialog in comparison to the mapping dialog. This is because our assumption is that the question terms which exist in the graph (OCs) should be interpreted before those which do not (POCs). Note that FREyA does not attempt to interpret the whole question at once, but it does it for one pair of OCs at the time. In other words, one resolved dialog can be seen as a pair of two OCs: an OC to which a question term is mapped, and the neighbouring OC (context). Secondly, the way the suggestions are generated for the two types of dialogs differ. The disambiguation dialog includes only the suggestions with Ontology Concepts that are the result of the ontology-based lookup (unless it is extended using the *forceDialog* mode, see Section 2.1). The mapping dialog, in contrast, shows the suggestions that are found through the ontology reasoning by looking at the closest Ontology Concepts to the POC (the distance is calculated by walking through the parsed tree). For the closest OC X, we identify its neighbouring concepts which are shown to the user as suggestions. *Neighbouring concepts* include the defined properties for X, and also its neighbouring classes. *Neighbouring classes* of class X are those that are defined to be 1) the domain of the property P where range(P)=X, and 2) the range of the property P where domain(P)=X. Finally, the sequence of disambiguation and mapping dialogs themselves controlled differently for these two kinds of dialogs:

- *The disambiguation dialogs* are driven by the question *focus* or the *answer type*, whichever is available first: the closer the OC to be disambiguated to the question focus/answer type, the higher the chance that it will be disambiguated before any other. The question focus is the term/phrase which identifies *what the question is about*, while the answer type identifies the type of the question such as *Person* in the query *Who owns the biggest department store in England?*. The focus of this question would be *the biggest department store* (details of the algorithm for identifying the focus and the answer type are described in [3]). After all ambiguities are resolved the FREyA workflow continues to resolve all POCs through the mapping dialogs.
- *The mapping dialogs* are driven by the availability of the OCs in the neighbourhood. We calculate the distance between each POC and the nearest OC inside the parsed tree, and the one with the minimum distance is the one to be used for the dialog before any other.

After all OCs are disambiguated and no POCs remain to be resolved, the system proceeds to finding the answer. First, it identifies the *answer type*, and then

combines OCs into triples, which are then used to generate the SPARQL query. Unlike other approaches which start by identifying the question type followed by the identification of the answer type, our approach tries to interpret the majority of the question before it identifies the answer type. The reason for this is that in FREyA there is no strict adherence to syntax, and the approach heavily relies on the ontology-based lookup and the definitions in the RDF structure. Hence, it can only identify the answer type after all relevant mappings and disambiguations are performed. Note however, that there are cases when the answer type is identified before the whole question is interpreted, and in this case it is used to drive the remaining mappings, if any (as described above).

An important part of FREyA is its **learning mechanism**. Our goal is to *learn the ranking* of the suggestions shown to the user so that after sufficient training the system can automatically generate the answer by selecting the best ranked options. In order to make the model as generic as possible, we do not update our learning model per question, but per combination of a POC/OC and the neighbouring OC (context). We also preserve a function over the selected suggestion such as minimum, maximum, or sum (applicable to datatype property values). This way we may extract several learning rules from a single question, so that if the same POC/OC appears in the same context, we can reuse it. The algorithm has previously shown a good performance on the Mooney GeoQuery dataset improving the initial suggestion rankings by 6% on a random sample of 103 questions (see [4]).

An Example Figure 2 shows the syntax tree for the query *what is the population of New York*. As *New York* is identified as referring to both *geo:State* and *geo:City*, we first ask the user to disambiguate (see Figure 2 a.)). If he selects for example *geo:City*, we start iterating through the list of remaining POCs. The next one (*population*) is used, together with the closest OC *geo:City*, to generate suggestions for the mapping dialog. Among them there will be *geo:cityPopulation* and after the user select this from the list of available options, *population* is mapped to the datatype property *geo:cityPopulation* (see Figure 2 b.)). Note that if the user selected that *New York* refers to *geo:State*, suggestions would be different, and following his selection, *population* would probably be mapped to refer to *geo:statePopulation* as the closest OC would be *geo:State*.

2.1 Querying Linked Data with FREyA

FREyA can be easily ported to work with a different ontology, or a set of ontologies. It can either preload the ontologies into its own repository which is based on OWLIM[4], or connect to an already existing repository, which can be local or remote.

In order to perform the ontology-based lookup at the query processing time, FREyA requires extracting ontology lexicalisations, processing them, and adding them to an index. The extraction of ontology lexicalisations requires reading the whole repository through a set of SPARQL queries. The number of SPARQL

[4] `http://ontotext.com/owlim`

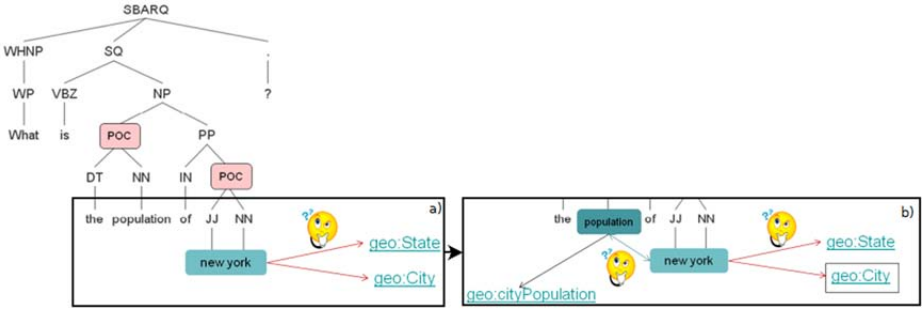

Fig. 2. Validation of potential ontology concepts through the user interaction

queries depends on the size of the schema which describes the dataset. Nowadays, the data are distributed over various types of servers, which often allow access through SPARQL endpoints. However, depending on the repository which is used underneath, some SPARQL queries can be highly unoptimised and slow. Alternative solution is to use services such as Watson [6] or Sindice [16] which index ontologies on the Web, in order to remove the burden of the system initialisation which can be large for large datasets. However, the downside of this approach is the lack of control over these services. As pointed out in [12], the resources on the open Web that can be accessed through Watson seem to have quality issues: there are many redundant, noisy and incomplete data (for example, the schema could be missing or ontologies might not be populated). These problems are partially addressed by approaches for assessing the quality through tracking the provenance (e.g.,[9]), however, much more needs to be done in the years to come in order to use and query the Web of Data effectively.

Most of the recently developed NLIs to ontologies (including our own QuestIO [5]) are built with the assumption that *ontologies are perfect*:

- each concept/relation in the ontology has the human lexicalisation which describes it – not necessarily a definition, but rather a term which a human would use to refer to this concept/relation;
- each concept/relation is positioned carefully in the taxonomy: super-concepts and super-relations are more generic, and sub-concepts and sub-relations are more specific.

The availability of Linked Open Data changed this assumption as encouraging people to publish their data resulted in the large amount of RDF graphs being made available and interlinked with each other. None of these are perfect – lexicalisations do exist, but not often they reflect "a term which a human would use to refer to a concept". In addition, the flat structure is dominant. One of the reasons for this is scalability: tractable reasoners do not scale well if the structure of the ontology is complex. The assumptions based on which NLIs to ontologies have been developed had to adapt to the new challenges, the main one being that *ontologies are not perfect*, and that tools which work with them must

take this into consideration. In addition, the *scale* becomes an issue, and also *incompleteness, heterogeneity, and noise* inherent in these data. A huge number of ontologies interlinked with each other means a high probability that there is a redundant information, which needs to be filtered out by the systems querying these data.

FREyA does not require a strict adherence to syntax, however, it relies on the ontology-based lookup. Trying a sample query *What is the capital of France?* with FREyA initialised with a superset of DBpedia (accessed through `http://www.factforge.net/sparql` repository) revealed that according to the extracted lexicon, **each word in the question refers to at least one Ontology Concept**. If there were no automatic disambiguation nor heavy grammar analysis, the system would model the first dialog asking: *What is 'what'? Is 'what' related to: LIST OF URIs.* A similar dialog would be modelled for 'is': the system would ask the user whether *is* is related to: *be*, *was*, or *were*. And so on, for each word in the question. These situations must be resolved either by performing automatic disambiguation (which might be expensive for datasets with billions of triples) or by constraining the supported language and allowing the user to type in only a limited set of question types. In case of the system failing to automatically interpret the question, it can prompt the user with the dialog as is the case with FREyA. The fine balance is in the combination of these approaches: disambiguate as much as possible and use the ranking mechanisms (e.g., those that exist in FREyA, or any other methods for effective ranking such as [13]), and correct them if necessary using the interactive features of FREyA.

When trying any dataset with FREyA for the first time, it is advisable to use the dialog as much as possible in order to check the system interpretations and correct them if necessary. In that regard, there are several *modes* that can be used, among which the most important are:

- **Automatic mode.** The system will simulate selection of the best ranked option(s) for each attempt to map a question term to an OC. This mode is used when the confidence is high that the ranking is effective, or the system has been trained enough and can make the decisions correctly. For the example previously described in Figure 2, the automatic mode would return both *statePopulation of new york state* and *cityPopulation of new york city* in the results initially as the initial ranking would assign the equal score to both*new york city* and *new york state*.
- **ForceDialog mode** generates the dialog for each attempt to map a question term to an OC.

The mode can be changed easily and without the need to reinitialise the system hence if the user uses FREyA in the automatic mode and discovers non-satisfying results, he can immediately switch to the *force dialog mode* in order to investigate the mappings. His input will then improve the system for the next user. Note that for the true ambiguities the automatic mode might not be the best choice even in the perfectly trained system. For instance, if somebody asks about *How big is new york state?* we might be unable to decide whether *How big* refers to *state area* or *state population* automatically. In this situation, as the system

learns from the user's suggestions, the automatic mode would work in favour of majority of the users. However, if the majority of users refer to *state area* when talking about size, the minority still have chance to get the correct answer by using FREyA in the *forceDialog* mode and mapping *big* to *state population*.

3 Evaluation

In this section we report the performance of FREyA using the MusicBrainz and DBpedia datasets provided within the QALD-1 challenge. We preloaded the data into our local repository (BigOWLIM 3.4, on the top of Sesame[5]) and then initialised the system using the SPARQL queries. Another option was to connect to the SPARQL endpoint provided by the QALD-1 challenge organisers[6], however, this was a difficult path due to the limited server timeout, which was not sufficient for executing all required queries.

Generating the index which is required for performing the ontology-based lookup is a mandatory step but is done once per dataset, although it might be time-consuming depending on the size of the data. Table 1 shows the statistics of loading the two datasets into the OWLIM repository and generating the index.

Table 1. Initialisation of the system and the size of datasets

	MusicBrainz	**DBpedia**
#explicit statements	14 926 841	328 318 709
#statements	19 202 664	372 110 845
#entities	5 490 237	96 515 478
#SPARQL queries executed	30	361623
initialisation time	1380s (0.38h)	182779s (50.77h)

After the index is generated, it is used at the query execution time. We first ran 50 training queries for both datasets and measured the overall precision, recall and f-measure. We then repeat the process with 50 test questions for each dataset. This experiment was conducted with FREyA in the *forceDialog* mode. Results are shown in Table 2[7]. MusicBrainz was a challenging dataset due to the existence of properties *beginDate* and *endDate*, which do not have any domain defined, and moreover, which are used extensively throughout the ontology and especially in the combination with the blank nodes. Several failures were due to the misfunction of the *Triple Generator* when these two properties were mapped to the wrong entity. For example, *Since when is Tom Araya a member of Slayer?* resulted in generating the following triples:

```
?joker1 - beginDate - Tom Araya (Artist)
Tom Araya (Artist) - member of band - ?joker2
?joker2 - toArtist - Slayer (Artist)
```

[5] `http://openrdf.org`

[6] `http://greententacle.techfak.uni-bielefeld.de:5171/sparql`

[7] Demos showing FREyA answering the QALD-1 challenge questions are available from `http://gate.ac.uk/sale/dd/`

Table 2. Performance of FREyA using QALD-1 datasets: the left figures exclude while the right figures include the questions correctly answered after reformulation. The number of dialogs per question includes only the questions that could be answered *correctly* with or without reformulation. *Not supported* questions include those that could not be correctly mapped to the correct SPARQL query due to the limited language coverage. For example, questions requiring negation, temporal reasoning such as *Which bands were founded in 2010?* or quantification such as in *Which locations have more than two caves?*. Partially correct questions are those that have returned a portion or a superset of the correct results.

	MusicBrainz		**DBpedia**	
	Training	*Testing*	*Training*	*Testing*
Precision	0.75/0.77	0.66/0.8	0.74/0.85	0.49/0.63
Recall	0.66/0.68	0.54/0.66	0.58/0.66	0.42/0.54
F-measure	0.70/0.74	0.59/0.71	0.67/0.72	0.45/0.58
# questions not supported	6	9	11	7
# reformulated questions	1	6	4	6
avg.#dialogs per question	3.4	3.65	2.7	2.85
# partially correct questions	1	1	3	12

and the corresponding SPARQL resulted in retrieving the birthday of Tom Araya, and not the date when he joined the group which is the correct answer.

Other challenges related to the ontology design in MusicBrainz include existence of the property *trackList* which has a container of type *rdf:Seq* as range. In addition, the statements with *releaseType* property use subclasses of class *Type* and not instances of that class which caused several failures. For example, the question *Who is the creator of the audiobook the Hobbit?* requires retrieving instances with lexicalisation *the Hobbit*, which are at the same time related to the class *TypeAudiobook* using the *releaseType* property, while FREyA expects that they are related using the *rdf:type* relation.

The main challenge with DBpedia was a selection of the property to use, due to the large number of suggestions that have always been present. For example, *Who created English Wikipedia?* could be mapped to *?joker dbp:created dbpedia:English_Wikipedia* while the correct answer is returned only after using *dbo:author* relation, instead of *dbp:created*[8]. In addition, there are many quality issues such as in the question *Who designed the Brooklyn Bridge?* where *designed* was mapped to *dbp:architect* instead of *dbp:designer* which resulted in retrieving `http://dbpedia.org/resource/John_Augustus_Roebling`, while using *dbp:designer* the result is `http://dbpedia.org/page/John_A._Roebling`. However, as no mapping exist between the two URIs, the former URI is not the same as the latter, and hence this is marked as an incorrect answer. Interestingly, the former URL is redirected to the latter, which indicates that the two URIs should also be connected using the property *sameAs* in the dataset.

[8] We use *dbp for* `http://dbpedia.org/property` and *dbo* for `http://dbpedia.org/ontology` namespaces.

Another challenge specific to DBpedia was the lack of the domain and range classes for properties. Therefore, some questions could not be correctly mapped to the underlying Ontology Concepts. In some cases, the reformulation of queries could help (such as using *spouse* instead of *married to*). However, reformulation was not always sufficient. For example, in *Which states border Utah?*, *border* needs to be mapped to the eight properties: *dbp:north*, *dbp:south*, *dbp:east*, *dbp:west*, *dbp:northwest*, *dbp:northeast*, *dbp:southwest*, and *dbp:southeast*. As none of these have any domain or range, they did not appear in the suggestions and hence the only way to answer the question using FREyA is to ask eight questions such as *Which states are north of Utah?*, *Which states are south of Utah*, and so on for each property. It is interesting to observe that 12 incorrectly answered questions using the DBpedia test questions were indeed partially correct. The correct mappings could only be placed if we were more familiar with the knowledge structure inherent in the dataset. This also explains the difference in the performance of FREyA using the training and the testing set of DBpedia.

Failures that were common for both datasets are related to the equal treatment of the datatype property values. For example, the question *How many jazz compilations are there?* failed to be answered correctly due to FREyA finding all compilations that had the user defined tag 'jazz' which is case insensitive (using FILTER REGEX(str(?var), "`^jazz$`","i"). Therefore, it included also 'Jazz' which lead to the incorrect answer. On the other hand, some entries were missed when the fuzzy matching was necessary such as in *Which companies are in the computer software industry?* that requires finding not only companies with the property *industry* 'computer software' but also 'computer hardware, software', 'computer software and engineering', and the like. At the moment, the datatype property values in FREyA are supported by including the exact match (case insensitive) only. In future, we might extend our approach to support more sophisticated treatment of strings so that the treatment differs depending on the context.

Several reformulations for both datasets resulted in a significant increase of the precision and recall, e.g. adding quotes such as in *Which artists performed the song "Over the Rainbow?"*. Without quotes, *Over* was parsed as a preposition, and the whole question failed to be answered, while with quotes this was a part of the Noun Phrase which lead to the correctly answered question.

Learning. To measure the effect of the learning mechanism, we run the experiment in two iterations: we first answered 50 testing questions using an empty learning model and then using the system trained with 50 training questions. Results are shown in Table 3.

The learning mechanism improved the overall ranking of suggestions for 0.05 for MusicBrainz, and only 0.02 for DBpedia. The reason is the size of the datasets and the relatively small number of the training questions. However, improvement of 0.02 is still an achievement considering that DBpedia has almost 100 million entities.

Table 3. Mean Reciprocal Rank for the testing set with and without learning

	MusicBrainz		**DBpedia**	
	untrained system	*trained system*	*untrained system*	*trained system*
MRR	0.63	0.68	0.52	0.54

Execution time for queries that could be answered correctly fluctuates based on the complexity of questions (e.g. number of the required dialogs). This is due to our on fly mechanism for finding suggestions which requires executing a large number of SPARQL queries in order to generate a dialog. Long execution is also affected by the complexity of the final generated SPARQL which is used to retrieve the answer. For example, queries which include FILTER statements over literal strings such as FILTER (regex(?var, "`^jazz$`", "i")) currently can take more than ten minutes to be executed[9]. The size of the dataset influences the execution time as well. For MusicBrainz, the average time per dialog was in the range from 0.073 to 11.4 seconds, or 8.5 seconds on average per question. For DBpedia, the execution time was much longer: from 5 to 232 seconds per dialog, and 36 seconds on average per question. This is quite slow, however, it can be optimised (e.g. by using the caching mechanisms for suggestions).

4 Conclusion and Future Work

We discussed the requirements and suitability of the Natural Language Interface – FREyA, to be used with different ontologies and for querying the Linked Open Data. The evaluation using the DBpedia and MusicBrainz testing datasets leads to the f-measure of 0.58 and 0.71 respectively which favourably compares to the other tested systems that participated in the QALD-1 challenge (PowerAqua 0.5 using DBpedia, SWIP 0.66 using MusicBrainz). More importantly, FREyA was the only system that is tested with both MusicBrainz and DBpedia datasets which demonstrates the portability. The learning mechanism improved the results for 5% and 2% for the MusicBrainz and DBpedia datasets respectively.

Acknowledgments. Authors would like to thank Ivan Peikov from Ontotext who helped with the configuration of OWLIM which was necessary for performing the experiments reported in this paper. This research is partially supported by the EU-funded LarKC (FP7-215535) project.

References

1. Cimiano, P., Haase, P., Heizmann, J.: Porting natural language interfaces between domains: an experimental user study with the orakel system. In: IUI 2007: Proceedings of the 12th International Conference on Intelligent User Interfaces, pp. 180–189. ACM, New York (2007)

[9] Experiments are run using the CentOS 5.2 Linux virtual machine running on a AMD Opteron 2431 2.40GHz CPU with 2 cores and 20G RAM.

2. Damljanovic, D.: Towards Portable Controlled Natural Languages for Querying Ontologies. In: Rosner, M., Fuchs, N. (eds.) Second Workshop on Controlled Natural Language. CEUR Workshop Pre-Proceedings, Marettimo Island, Italy, September 13-15, vol. 622 (2010) ISSN 1613-0073, http://ceur-ws.org
3. Damljanovic, D., Agatonovic, M., Cunningham, H.: Identification of the Question Focus: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction. In: 7th Language Resources and Evaluation Conference (LREC), ELRA, La Valletta, Malta (May 2010)
4. Damljanovic, D., Agatonovic, M., Cunningham, H.: Natural Language Interfaces to Ontologies: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010. LNCS, vol. 6088, pp. 106–120. Springer, Heidelberg (2010)
5. Damljanovic, D., Tablan, V., Bontcheva, K.: A text-based query interface to owl ontologies. In: 6th Language Resources and Evaluation Conference (LREC). ELRA, Marrakech (2008)
6. d'Aquin, M., Baldassarre, C., Gridinoc, L., Angeletou, S., Sabou, M., Motta, E.: Watson: A Gateway for Next Generation Semantic Web Applications. In: Poster, ISWC 2007 (2007), http://iswc2007.semanticweb.org/papers/Paper366-Watson-poster-ISWC07.pdf
7. Fellbaum, C. (ed.): WordNet - An Electronic Lexical Database. MIT Press (1998)
8. Grosz, B.J., Appelt, D.E., Martin, P.A., Pereira, F.C.N.: TEAM: An experiment in the design of transportable natural-language interfaces. Artificial Intelligence 32(2), 173–243 (1987)
9. Hartig, O., Zhao, J.: Using Web Data Provenance for Quality Assessment. In: Proceedings of the First International Workshop on the Role of Semantic Web in Provenance Management (SWPM 2009) at the International Semantic Web Conference (ISWC 2009), Washington D.C., USA (2009)
10. Kaufmann, E., Bernstein, A., Fischer, L.: NLP-Reduce: A Naive but Domain-independent Natural Language Interface for Querying Ontologies. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 1–2. Springer, Heidelberg (2007)
11. Klein, D., Manning, C.D.: Fast exact inference with a factored model for natural language parsing. In: Becker, S., Thrun, S., Obermayer, K. (eds.) Advances in Neural Information Processing Systems, NIPS 2002, vol. 15, pp. 3–10. MIT Press (2002), http://books.nips.cc/papers/files/nips15/CS01.pdf
12. Lopez, V., Fernández, M., Motta, E., Stieler, N.: PowerAqua: Supporting Users in Querying and Exploring the Semantic Web Content. Semantic Web Journal (2011)
13. Lopez, V., Nikolov, A., Fernandez, M., Sabou, M., Uren, V., Motta, E.: Merging and Ranking Answers in the Semantic Web: The Wisdom of Crowds. In: Gómez-Pérez, A., Yu, Y., Ding, Y. (eds.) ASWC 2009. LNCS, vol. 5926, pp. 135–152. Springer, Heidelberg (2009), http://dx.doi.org/10.1007/978-3-642-10871-6_10
14. Lopez, V., Uren, V., Motta, E., Pasin, M.: Aqualog: An ontology-driven question answering system for organizational semantic intranets. Web Semantics: Science, Services and Agents on the World Wide Web 5(2), 72–105 (2007)
15. Lopez, V., Uren, V.S., Sabou, M., Motta, E.: Cross Ontology Query Answering on the semantic Web: an Initial Evaluation. In: Gil, Y., Noy, N.F. (eds.) K-CAP, pp. 17–24. ACM (2009)

16. Tummarello, G., Delbru, R., Oren, E.: Sindice.com: Weaving the Open Linked Data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 552–565. Springer, Heidelberg (2007)
17. Wang, C., Xiong, M., Zhou, Q., Yu, Y.: Panto: A Portable Natural Language Interface to Ontologies. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 473–487. Springer, Heidelberg (2007)