MuG-QA: Multilingual Grammatical Question Answering for RDF Data

Elizaveta Zimina, Jyrki Nummenmaa, Kalervo Järvelin, Jaakko Peltonen, and Kostas Stefanidis
University of Tampere
Finland
firstname.lastname@uta.fi

Abstract—We introduce Multilingual Grammatical Question Answering (MuG-QA), a system for answering questions in the English, German, Italian and French languages over DBpedia. The natural language modelling and parsing is implemented using Grammatical Framework (GF), a grammar formalism having natural support for multilinguality. The question analysis is based on forming an abstract conceptual grammar from the questions, and then using linearisation of the abstract grammar into different languages to parse the questions. Once a natural language question is parsed, the resulting abstract grammar tree is matched with the knowledge base schema and contents to formulate a SPARQL query. A particular strength of our approach is that once the abstract grammar has been designed, implementation for a new concrete language is relatively quick, supposing that the language has basic support in the GF Resource Grammar Library. MuG-QA has been tested with data from the QALD-7 benchmark and showed competitive results.

Index Terms—Grammatical Framework, DBpedia, question answering, QALD, multilinguality.

I. INTRODUCTION

Nowadays the rapid development of RDF knowledge bases require functionalities that support natural language queries to cover the users’ search needs. The amount of non-English data is increasing, as well as the number of users who want to use this data and make queries in various languages. These aspects are the focus of the Question Answering over Linked Data (QALD) workshops [1] organised over the last 8 years, and the related Scalable Question Answering (SQA) challenge [2].

In the current work, we present the Multilingual Grammatical Question Answering (MuG-QA) system, to answer questions in several natural languages. This is a multilingual extension of the GQA system for the English language, described in detail in [3]. MuG-QA is based on a multilingual controlled language grammar built by means of Grammatical Framework (GF) [4], thus bringing the capabilities of the grammatical approach and GF’s multilingual power to question answering. GF enabled us to map concepts across languages relatively easily, saving time and effort in adding new languages to the multilingual system. We implement the system based on DBpedia as the data store and evaluate it on the QALD-7 benchmark (Task 1) with competitive results.

This work has been supported by the Academy of Finland, decision number 313748, and by the Virpa D project funded by Business Finland.
Number = Sg | Pl ;
Gender = Masc | Fem ;

and then define the linearisation types for \( A \) and \( N \):
\[
A = \{ s \rightarrow \text{Gender} \rightarrow \text{Number} \rightarrow \text{Str} \} ;
\]
\[
N = \{ s \rightarrow \text{Number} \rightarrow \text{Str} ; g \rightarrow \text{Gender} \} ;
\]

String values \( s \) of both \( A \) and \( N \) are represented by inflectional tables, whereas \( N \) has a value for gender. Now, the linearisation definition for the rule \( \text{mkNP} \) can be written:
\[
\text{mkNP} \text{adj noun} = \{ s = “\n” \rightarrow \text{noun.s} \rightarrow n + + \text{adj.s} \rightarrow \text{noun.g} \rightarrow n ; g \rightarrow \text{noun.g} \} ;
\]

Our linearisation function \( s \) producing the string value of \( \text{NP} \) is obtained by joining the generated string values of \( A \) and \( N \) so that the adjective agrees with the noun in gender and number, and the value \( g \) is inherited from the noun. The lambda notation “\n” shows that the \( s \) value of \( \text{NP} \) and \( N \) are both tables (\( \text{Number} \rightarrow \text{Str} \)).

We now add two vocabulary rules to the abstract syntax:
\[
\text{signora}_N : N ;
\]
\[
\text{italiano}_A : A ;
\]

and their linearisation definitions to the concrete syntax:
\[
\text{signora}_N = \{ s = \text{table} \{ \text{Sg} \rightarrow “signora”; \text{Pl} \rightarrow “signore” \} ; g = \text{Fem} \} ;
\]
\[
\text{italiano}_A = \{ s = \text{table} \{ \text{Masc} \rightarrow \text{table} \{ \text{Sg} \rightarrow “italiano”; \text{Pl} \rightarrow “italiani” \} ; \text{Fem} \rightarrow \text{table} \{ \text{Sg} \rightarrow “italiana”; \text{Pl} \rightarrow “italiane” \} \} \} ;
\]

The parse tree of the string \( \text{signora italiana} \) looks like:
\[
\text{mkNP} \text{italiano}_A \text{signora}_N .
\]

The \( \text{mkNP} \) rule takes care of agreement and word order.

We can write the linearisation rules and definitions for other languages similarly. They will follow the morphological rules of concrete languages, but at the same time will be related to the shared abstract categories and rules.

B. DBpedia

The MuG-QA system is based on the DBpedia version 2016-04 [7]. The main KB components involved in the search include entity labels, redirect and disambiguation pages, ontology and infobox statements (\( \text{entity} \rightarrow \text{property} \rightarrow \text{value} \) triples), and ontology types of entities (e.g. \( \text{city} \), \( \text{animal} \), \( \text{game} \), etc.). In case of ontology and infobox statements, we also implemented inverse indexes, enabling us to look for subjects in RDF triples, and not only values.

For questions in languages other than English, the system outputs answers as SPARQL on English DBpedia version. For this, we use the mappings of English entity labels on to their equivalents in German, Italian and French, as well as redirects and disambiguations in these languages (Table I).

<table>
<thead>
<tr>
<th>Language</th>
<th>Entities</th>
<th>Redirects</th>
<th>Disambiguations</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>6.0M</td>
<td>7.3M</td>
<td>1.5M</td>
</tr>
<tr>
<td>German</td>
<td>1.7M</td>
<td>1.3M</td>
<td>1.1M</td>
</tr>
<tr>
<td>French</td>
<td>1.6M</td>
<td>1.4M</td>
<td>542K</td>
</tr>
<tr>
<td>Italian</td>
<td>970K</td>
<td>570K</td>
<td>315K</td>
</tr>
</tbody>
</table>

III. THE MU-G-QA GRAMMAR

The MuG-QA grammar shares the abstract syntax of the GQA system described in [3]. Thus, we have previously considered the main conceptual categories of the grammar, such as entities, properties, classes, verb chunks, relative clauses and questions (as a starting category).

The novelty of MuG-QA is the concrete syntaxes for the German, Italian and French languages, operating with the same categories and concepts of the common abstract syntax. Their implementation was complicated by the linguistic peculiarities of the languages and, most importantly, the need to map names of entities, properties and classes onto their equivalents in the English language, since the system is tailored to give answers based on the English DBpedia.

The conceptual categories are the building blocks for various rules in the grammar. The example below shows the representation of a question through abstract syntax rules and a way of its interpretation.

A. Parse Example

Consider the parse of Does the Isar flow into a lake?:

\[
\text{DoesXP} \text{(twoWordEnt “the” “Isar”) (VPSlash_to_VPChunk flowInto_IVPS (Class_to_Ent (ArtAdjClassChunk (Class_Nom_Chunk Lake_CNClass))))}
\]

The top function \( \text{DoesXP} \) is declared in the grammar as

\[
\text{DoesXP} : \text{Entity} \rightarrow \text{VPChunk} \rightarrow Q, \text{ie. a function taking two arguments (Entity and VPChunk) to form a question.}
\]

In our case, \( \text{Entity} \) is made of two words (the Isar), which are used to find the link of the entity existing in DBpedia.

\[
\text{VPChunk} \text{is formed by the rule VPSlash_to_VPChunk : VPSlashChunk \rightarrow Entity \rightarrow VPChunk, involving, except Entity, the category VPSlashChunk that means any grammatical form of a verb phrase missing a complement (an analogue of VPSlash in the RGL, but grammatically independent). The prepositional verb flow into is learnt from the training set and coded as a VPSlashChunk; such verbs are called idiomatic VPSlash chunks (IVPS) in the MuG-QA grammar. To make a complete verb phrase out of these verbs, we need some Entity, which could be another set of symbols, e.g. Kochelsee, if the question was Does the Isar flow into Kochelsee? The MuG-QA grammar also allows classes to act as entities, so the class \( \text{lake} \) becomes an entity by means of three rules:
\]

\[
\text{Class Nom Chunk : CNClass \rightarrow NPClassChunk ;}
\]
ArtAdjClassChunk : NPClassChunk → NPClassChunk ;
Class_to_Entity : NPClassChunk → Entity.

In terms of morphology, the function Class_Nom_Chunk comprises all forms of a CNClass (common noun class) in the nominative case. For example, in English it means that it would match both singular and plural numbers (lake and lakes). ArtAdjClassChunk makes it possible to attach any article or some words and word combinations to the class (e.g. famous, some, one of, etc.) and to stay within the same category NPClassChunk. The last function turns it into Entity.

One could notice that the MuG-QA grammar involves “chunks”, that is, grammatically independent elements, which leaves a possibility for ungrammatical constructions. For example, the above-mentioned parse would also accept the verb form flows and the noun phrase a lakes. This makes it possible to tolerate mistakes in questions and still comprehend their meaning. At the same time, this autonomy of constituents makes parses shorter and easier to read for humans. A MuG-QA parse is aimed to present a conceptual structure of a question, rather than to focus on certain linguistic nuances.

In the following subsection we will closer look at the grammar categories, rules forming them and techniques of finding DBpedia information that they represent for languages other than English.

B. Conceptual Categories in the German, Italian and French Languages

1) Simple Entities: To find the links of entities in the English DBpedia given their names in other languages, we use the Interlanguage Links dataset:

http://dbpedia.org/resource/Wales →
http://de.dbpedia.org/resource/Wales
http://it.dbpedia.org/resource/Galles
http://fr.dbpedia.org/resource/Pays_de_Galles

Redirects and disambiguations are used for each language:
Pierre Strouve → http://fr.dbpedia.org/resource/Pierre_Strouve
Foro → http://it.dbpedia.org/resource/Foro_(tribunale)
http://it.dbpedia.org/resource/Foro_Romano
http://it.dbpedia.org/resource/Foro_(fiume)

Sometimes (especially in non-English DBpedia versions) the entity page is found, but it contains no information relevant to the question. For example, the entity indaco (“indigo”) in the Italian-language question Qual la lunghezza d’onda dell’indaco? (“What is the wavelength of indigo?”) is directly linked to the English page http://dbpedia.org/resource/Indigo_dye, whereas the relevant link http://dbpedia.org/page/Indigo is related to the Italian page http://it.dbpedia.org/resource/Indaco_(colore). Thus, it is necessary to do a deeper search among disambiguation pages if the immediate page seems to be irrelevant.

2) Properties: Since answering questions in languages other than English is based on the English DBpedia, we need to correlate English names of properties with their equivalents in other languages. Unlike entities, DBpedia properties do not have sameAs links to their URIs in different languages. Therefore, we translated English labels of properties into our target languages using Google Translate. For the most common DBpedia properties machine translation works correctly, since labels are usually short expressions whose translations are often checked by the Translate Community. Thus, in the MuG-QA grammar, property translations are used in linearisation definitions of the same rules from the abstract syntax, so that, for example, the parse of the sentence Donnez-moi le lieu de naissance de Frank Sinatra (“Give me the birth place of Frank Sinatra”) will contain the function birthPlace_O, which will be the same for other languages as well.

3) Classes: As well as for the English language, classes in other concrete grammars are considered as common nouns (CNs), and their morphological flexibility is provided by the RGL paradigms. For example, in the MuG-QA grammar the chemical substance class is represented with the tree AdjCN(PluralA chemical_A)(UseN substance_N).

The linearisations of chemical_A and substance_N are taken from the RGL wide-coverage dictionary. Implemented for all languages, the RGL function AdjCN takes care of the word order and agreement between its elements (It.: sostanza chimica, not chimico or chimici). Thus, the parse of a question with this class in any language would contain the ChemicalCompound_CNClass function, referring us to the URI http://dbpedia.org/ontology/ChemicalCompound. In this way, the RGL makes it simpler to focus on the semantic structure of a question, without a great effort in grammar analysis.

4) Verb Phrases: Similar to classes, most verbs and verb phrases do not need special implementation in the concrete grammars for the languages other than English. The corresponding rules just should be copied from the wide-coverage dictionary, and the system will recognise all the conjugation forms of those verbs. For example, écrit, écrivent, écrivait, écrivai, etc in French and schreibt, schreiben, schrieb, schrieben, etc in German will be attributed to the grammar rule write_V2, due to the verb paradigms inherited from the RGL grammar.

IV. Testing

The MuG-QA conceptual grammar was built with the QALD-7 Task 1 (215 questions) and (partly) SQA (5000 questions) training sets and the system was then tested on the QALD-7 Task 1 test set. This task is focused on factual, mostly short questions formulated in different languages (English, German, Italian and French), which should be answered through SPARQL queries.

1https://translate.google.com/
The challenge outcome overview [8] reports that the size of the test set was 50 questions, however we could obtain only 43 questions from the current webpage [9]. The public test set does not contain original answers, so we recreated them manually. We evaluated the results of our system over 43 questions in the public test set (Table II). We consider a question as “parsed” if the system produces a correct parse of it, “processed” if the system gives some output SPARQL query, and an answer as “correct” if this SPARQL query produces the answer coinciding with the recreated one.

The reported results of the challenge are presented in Table III. In QALD-7, participant systems were evaluated by microaveraged precision and recall (overall ratio of the number of returned correct results to number of all returned results or to number of all correct results, respectively) and by macroaveraged precision and recall (averages of per-question precision and recall measures over the questions), and corresponding microaveraged and macroaveraged F1-measures. Unanswered questions were left out of the averages for the macroaveraged measures [8]. Following the same evaluation scheme, information retrieval measures for MuG-QA are presented in Table II: this information retrieval evaluation is the closest equivalent to the QALD-7 evaluation available for the public data and the results are roughly comparable to the results in Table III. MuG-QA yields a perfect 1.00 score for macroaveraged precision, macroaveraged recall, macroaveraged F1, and microaveraged precision on all languages; this is because whenever MuG-QA processed the query, the answers coincided with the correct ones. Under a simplifying assumption that the average number of correct answers per question is the same in unanswered questions as in answered ones, microaveraged recall of MuG-QA becomes the same as Accuracy in Table II.

The results show that MuG-QA performs better for English than the best previous English-language system of QALD-7, by all evaluation measures. For Italian and German MuG-QA also received very good scores. French had the worst performance, although MuG-QA still yields better macroaveraged results than the previous results for French.

The analysis of our results shows that the sets of questions that were answered correctly are with a few exceptions the same among the four languages. This is an expected outcome, since the grammars for all languages were based on the same training sets and were organised in the same way. However, the performance in Italian, German and French turned out to be lower than the one for English due to some inconsistencies in the correlation of grammar components or differences in question formulations.

For example, the English, Italian and German grammars successfully “coped” with the question Give me all chemical elements. French for this question was Donnez-moi le nom de tous les iments chimiques (“Give me the name of all the chemical elements”). As the phrase le nom was not in the French training set, the question could not be parsed.

Another challenge is translation of properties. For example, runtime in the test sentence Give me the runtime of Toy Story was properly translated only into German (Laufzeit), while in Italian and French Google Translate preferred to leave it as runtime instead of durata and dure accordingly.

Some questions revealed the problems in correlating entity labels across languages. Thus, the Interlanguage Links, Redirects and Disambiguations datasets for French did not help the system to understand that courses hippiques in the question Est-ce que les courses hippiques sont un sport? (“Is horse racing a sport?”) should be attributed to horse racing, whose analogue in French DBpedia is sport Hippique.

MuG-QA is also not taught to perform any mathematical operations, e.g. the question How big is the earth’s diameter? could not be answered, since the entity Earth in DBpedia has the property mean radius, not diameter, and cases of this kind have not been met in the training sets.

Testing proved our assumption that the main shortcoming of MuG-QA is its focus on the controlled language, so that it cannot process previously unseen question formulations. At the same time, QALD-7 task 1 is mostly oriented on semantically varied, manually created questions, unlike SQA with its emphasis on complex syntactic structures. As noted, however, MuG-QA performed well compared to previous QALD-7 systems.

Our system could be improved by methods that tolerate broader variation of question wording, e.g. lexical matching of DBpedia properties and question tokens excluding entity names, which was implemented in [10]. At the same time, their method makes it possible to answer only simple questions, concerning one named entity and its single property.

V. RELATED WORK

Three systems officially participated in solving QALD-7 task 1 in 2017 [8]: WDAqua, qanswer2, and AMAL. WDAqua [11] is a rule-based system transforming natural language questions into SPARQL queries by means of the combinatorial approach, supporting English on DBpedia and English, French, German and Italian on Wikidata. qanswer2 [12] produces semantic query graphs and looks for matches of questions in the generated subgraphs, at the same time resolving ambiguities. AMAL [10] answers questions in French through question classification, extraction of entities and semantic matching of the remaining part with DBpedia properties, involving a manually built lexicon.

Plenty of systems have been developed outside the QALD challenge, involving various KBs: DBpedia [13], Freebase [14]–[16], WikiAnswers [14], ClueWeb [14], [17], WebQuestions [14], ReVerb [18], MusicBrainz [19], etc.

The MuG-QA system is a multilingual extension of GQA [3], that is, it exploits the same categories and syntax, but covers three other languages in addition to English – German, Italian and French – and has the potential of increasing this number with low effort due to the multilingual power of the Grammatical Framework [4]. Unlike most related question answering systems based on a controlled natural language [20], [21], the MuG-QA has the advantage
TABLE II
PERFORMANCE OF MuG-QA OVER THE QALD-7 TASK 1 PUBLIC TEST SET (43 QUESTIONS)

<table>
<thead>
<tr>
<th>Language</th>
<th>Parsed, Processed, Correct</th>
<th>Accuracy</th>
<th>Micro and Macro Precision</th>
<th>Macro Recall and F1</th>
<th>Micro Recall</th>
<th>Micro F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>22, 22, 22</td>
<td>0.512</td>
<td>1.000</td>
<td>1.000</td>
<td>0.512</td>
<td>0.677</td>
</tr>
<tr>
<td>French</td>
<td>19, 17, 17</td>
<td>0.395</td>
<td>1.000</td>
<td>1.000</td>
<td>0.395</td>
<td>0.566</td>
</tr>
<tr>
<td>Italian</td>
<td>22, 21, 21</td>
<td>0.488</td>
<td>1.000</td>
<td>1.000</td>
<td>0.488</td>
<td>0.656</td>
</tr>
<tr>
<td>German</td>
<td>20, 19, 19</td>
<td>0.442</td>
<td>1.000</td>
<td>1.000</td>
<td>0.442</td>
<td>0.613</td>
</tr>
</tbody>
</table>

TABLE III
PERFORMANCE OF THE PARTICIPATING SYSTEMS OVER THE QALD-7 TASK 1 TEST SET (50 QUESTIONS) [8]

<table>
<thead>
<tr>
<th>System</th>
<th>WDAqua</th>
<th>ganswer2</th>
<th>AMAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language</td>
<td>English</td>
<td>English</td>
<td>French</td>
</tr>
<tr>
<td>Macro Precision</td>
<td>0.080</td>
<td>0.322</td>
<td>0.998</td>
</tr>
<tr>
<td>Micro Recall</td>
<td>0.006</td>
<td>0.127</td>
<td>0.989</td>
</tr>
<tr>
<td>Micro F1-measure</td>
<td>0.012</td>
<td>0.182</td>
<td>0.993</td>
</tr>
<tr>
<td>Macro Precision</td>
<td>0.162</td>
<td>0.487</td>
<td>0.720</td>
</tr>
<tr>
<td>Micro Recall</td>
<td>0.160</td>
<td>0.498</td>
<td>0.720</td>
</tr>
<tr>
<td>Macro F1-measure</td>
<td>0.143</td>
<td>0.469</td>
<td>0.720</td>
</tr>
</tbody>
</table>

of being fully automatic, since it does not involve manual reformulation of questions into some required format.

VI. CONCLUSION

MuG-QA is a DBpedia-based question answering system supporting systematically several, currently four languages. Its important benefit is a relatively easy and fast way of adding new languages to the system. The abstract syntax is the same for all languages, whereas concrete syntaxes are analogous with each other, and the morphology is mostly inherited from the Grammatical Framework’s resource grammars, reducing the amount of manual work to a minimum.

The approach involving controlled natural language increases the probability of understanding questions accurately and retrieving correct answers. Arguably, one would never foresee all possible formulations of natural language questions, even using a large training set. Nevertheless, our approach does not require similar manual reformulation of questions as in other controlled language systems.

Our system showed competitive results. It can be further developed particularly by improving its semantic flexibility and by adding more languages.

REFERENCES