Generating Grammars from lemon lexica for Questions Answering over Linked Data: a Preliminary Analysis

Viktoria Benz[1], Philipp Cimiano[1], and Mohammad Fazleh Elahi[1] Basil Ell[1]

[1] Semantic Computing Group, CITEC, Universität Bielefeld, Inspiration 1, 33619 Bielefeld, Germany
[2] {vbenz,cimiano,melahi,bell}@techfak.uni-bielefeld.de

Abstract. Most approaches to question answering over linked data (QALD) frame the task as a machine learning problem, consisting in learning a mapping from natural language questions into SPARQL queries by parametrizing a model from training data given in the form of pairs of natural language (NL) question and SPARQL query. In this preliminary work we present an alternative approach to developing a QA system using machine learning that relies on the automatic generation of a QA grammar from a lemon lexicon. This model-based approach comes with a number of advantages compared to a machine learning approach. First, our approach gives maximum control over the QA interface to the developer of the system as every entry added to the lexicon increases the coverage of the grammar and thus of the QA system in a predictable way. This is in contrast to machine learning approaches where the impact of the addition of a single training example is difficult to predict. A further advantage of our approach is that the QA system operates on the basis of a symbolic grammar that can be used to provide guidance and auto-completion functionality to users. Our system is indeed intended to be used in the context of an auto-completion interface that allows users to ask only questions that the grammar can cover. We present very preliminary results showing that a large percentage of the questions of the training set of QALD-7 can be rephrased in terms of questions that our grammar can parse. We show that with a hand-crafted lexicon, we can in principle get very high micro-F1 scores of 62.5% on the training data of QALD-7 when questions are manually rephrased to fit our grammar. Although these preliminary results do not constitute a proper evaluation of our approach, they hint at the fact that an approach as we propose seems feasible.

Keywords: grammar generation, question answering over linked data, lemon

1 Introduction

Most approaches to question answering over linked data (QALD) follow a machine learning (ML) approach where a QA system is learned by parametrizing a model on the basis of training data given in the form of pairs of NL question and SPARQL query. In this preliminary work, we explore an alternative paradigm that consists in generating a lexicalized grammar that can be used to parse questions into SPARQL. The method relies on the availability of a lemon lexicon [18] for the given ontology / knowledge graph that is queried. In our view this does not represent a limitation as the lexicon can be also used in other tasks, e.g. to verbalize the ontology. Further, the approach of developing such lexica can be scaled up by a collaborative approach [17].
An advantage of our proposed paradigm is that it provides a level of control over the QA interface that ML methods can not provide. In fact, for ML systems in general, it is unclear what the impact of adding one example is in terms of behaviour of the system. Generally, several examples of the same or similar type need to be provided for the ML system to learn a pattern, leading to high redundancy in training data. In contrast, in our model-based approach, several lexicalized grammar rules are generated automatically for each entry in the lemon lexicon and the impact of each lexicon entry on the behaviour of the QA system can be clearly predicted and monitored. A further limitation of machine learning approaches is that they can not be directly used in a guided interface that offers auto-completion features as this requires access to a grammar or a correspondingly powerful look-ahead. Finally, adopting a QA system to different domains requires the creation of training datasets with hundreds if not thousands of questions to train a system from scratch for the new domain or ontology. We shift this effort to the creation of a lexicon that can represent a cost-effective approach compared to creating large amount of redundant examples the impact of which is not clear a priori.
For these reasons, we have developed an approach that can automatically generate a lexicalized grammar from a lemon lexicon. The approach is inspired by earlier work on generating LTAG grammars [21]. Here we revisit this approach in the context of QALD interfaces and provide a preliminary evaluation of our approach that shows that, if a corresponding lexicon is available, our approach can reach micro F-1 values of *62.5%* on the English dataset of QALD-7[3] [22], provided that queries are rephrased with our grammar.
This paper is structured as follows: in the next Section we describe our approach and in particular how regular lexicalized grammars are generated from lemon lexica. These grammars can be used to parse questions and map them into corresponding SPARQL queries. We provide a preliminary feasi-

[3] http://qald.aksw.org/

bility study for our approach by evaluating the approach on QALD-7 training data. We manually rephrase questions in QALD-7 so that they can be analysed with our grammar. While the rephrasing might be seen as a critical limitation, we regard this rephrasing as legitimate given that we see the application of our grammar-based approach in the context of a guided NL interface with auto-completion functionality such as proposed by Rico et al. [19]. So our main question is not whether people would be able to ask the questions from QALD-7 as they are given, but whether they would be able to ask a question that satisfies their information need using our grammar.

2 Generating QA grammars from lemon lexica

Our approach automatically generates lexicalized regular grammars from lexical entries in a lemon lexicon and is inspired in previous work that showed how to generate LTAG grammars from a specification of the ontology-lexicon interface [21]. The grammar generation approach works for four basic types of entries corresponding to:

- relational nouns subcategorizing a prepositional phrase (NounPPFrame in Lexinfo [7]), e.g. *'capital (of)'*
- transitive verbs (TransitiveFrame), e.g. *(to) 'direct'*
- intransitive verbs subcategorizing a prepositional phrase, (IntransitivePPFrame), e.g. *'flow through'*
- adjectives (AdjectiveAttributiveFrame), e.g. *'spanish'*

We describe the grammar entries generated for each of these four types in the following subsections

2.1 NounPPFrame

Consider the lemon lexical entry in Figure 1 for the relational noun *'capital' (of)*. The entry states that the canonical written form of the entry is *"capital"*. It states that the entry has a NounPPFrame as syntactic behaviour that corresponds to a copulative construction *'X is the capital of Y'* with two arguments, where `copulativeArg` corresponds to the copula subject X and the `prepositionalAdjunct` corresponds to the prepositional object Y. The semantics of the relational noun is captured with respect to the property `http://dbpedia.org/ontology/capital`, where the subject of the property is realized by the `prepositionalAdjunct` and the object of the property is realized by the `copulativeArg`. This essentially captures the fact that the meaning of *'Berlin is the capital of Germany'* is expressed by the triple

```
<Germany> <http://dbpedia.org/ontology/capital> <Berlin>
```

```
:lexicon_en a lemon:Lexicon ;
  lemon:language "en" ;
  lemon:entry :capital_of ;
  lemon:entry :of .

:capital_of a lemon:LexicalEntry ;
  lexinfo:partOfSpeech lexinfo:noun ;
  lemon:canonicalForm :capital_form ;
  lemon:synBehavior :capital_of_nounpp ;
  lemon:sense :capital_sense_ontomap .

:capital_form a lemon:Form ;
  lemon:writtenRep "capital"@en .

:capital_of_nounpp a lexinfo:NounPPFrame ;
  lexinfo:copulativeArg :arg1 ;
  lexinfo:prepositionalAdjunct :arg2 .

:capital_sense_ontomap a lemon:OntoMap, lemon:LexicalSense ;
  lemon:ontoMapping :capital_sense_ontomap ;
  lemon:reference <http://dbpedia.org/ontology/capital> ;
  lemon:subjOfProp :arg2 ;
  lemon:objOfProp :arg1 ;
  lemon:condition :capital_condition .

:capital_condition a lemon:condition ;
  lemon:propertyDomain <http://dbpedia.org/ontology/Country> ;
  lemon:propertyRange <http://dbpedia.org/ontology/City> .

:arg2 lemon:marker :of .

:of a lemon:SynRoleMarker ;
  lemon:canonicalForm [ lemon:writtenRep "of"@en ] ;
  lexinfo:partOfSpeech lexinfo:preposition .
```

Fig. 1. Lemon entry for the relational noun *'capital (of)'*

The entry also captures that the default domain of the property is `http://dbpedia.org/ontology/Country` in the context of the lexical entry (other lexical entries might induce other defaults for the domain/range). Correspondingly, the default range of the property is `http://dbpedia.org/ontology/City`. From this lexical entry, two grammar rules are automatically generated in our approach.

The 1st grammar rule (Figure 2) is a rule for the English language that is of type "SENTENCE", that is, it 'generates' full sentences (in our case a full question). The rule is generated from an entry following the NounPPFrame. The grammar entry is lexicalized as it refers to specific lexical elements. It 'generates' the following four proto-questions: 1) *'What is the capital of X?'*, 2) *'What was the capital of X?'*, 3) *'Which city is the capital of X?'*, 4) *'Which city was the capital of X?'*. These are proto-questions in the sense that some elements need to be inserted at the X position to be a complete sentence or question. Two different types of elements can be inserted into the X position. On the one hand, we can insert labels denoting a particular country, e.g. *'Germany'*. This is referred to as $x in the grammar entry and any label of a country can be inserted into the position. On the other hand, a complete noun phrase (NP) denoting a country can be inserted, such as *'country where German is spoken'*, *'country were Einstein was born'*, *'country governed by Angela Merkel'*, etc. $x can be regarded as a preterminal symbol and COUNTRY_NP as a non-terminal symbol. The grammar entry de-

```
{
  "id": "107",
  "language": "EN",
  "type": "SENTENCE",
  "bindingType": "COUNTRY",
  "returnType": "CITY",
  "frameType": "NPP",
  "sentences": [
    "What is the capital of ($x | COUNTRY_NP)?",
    "What was the capital of ($x | COUNTRY_NP)?",
    "Which city is the capital of ($x | COUNTRY_NP)?",
    "Which city was the capital of ($x | COUNTRY_NP)?"
  ],
  "queryType": "SELECT",
  "sparqlQuery": "(bgp (triple ?subjOfProp <http://dbpedia.org/ontology/capital> ?objOfProp))",
  "sentenceToSparqlParameterMapping": {
    "$x": "subjOfProp"
  },
  "returnVariable": "objOfProp",
  "sentenceBindings": {
    "bindingVariableName": "$x",
    "bindingList": [{
      "label": "Democratic Republic of Afghanistan",
      "uri": "http://dbpedia.org/resource/Democratic_Republic_of_Afghanistan"
    }],...
  },
  "combination": false
}
```

Fig. 2. 1st grammar rule automatically generated from the entry in Figure 1

fines the semantics of the questions by way of basic graph patterns (bgps), in this case the pattern:

```
(bgp (triple ?subjOfProp <http://dbpedia.org/ontology/capital> ?objOfProp))
```

In the entry it is further specified that the fillers of the placeholder $x are to be substituted into the subjOfProp position, and the objOfProp represents the return variable of the query. The “sentenceBindings” element lists all the possible entities that $x can be bound to. This binding list is only abbreviated in the Figure and is obtained by querying the corresponding knowledge graph. The flag “combination: false” indicates that this is a grammar rule with the start symbol at the left side that can not be combined with another rule.

The 2nd grammar rule generated for the lemon entry in Figure 1 is given in Figure 3. This rule generates the noun phrase *‘the capital of $x’* where any country can be inserted into the $x position. The “returnType” is CITY, meaning that this noun phrase can be inserted into any other rule requiring a CITY_NP. The rest of the grammar rule is similar to the case above.

2.2 Transitive Verbs

The second example entry we discuss is the entry given for the verb *(to) ‘direct’* given in Figure 4. The lexical entry states that the canonical form has the written representation *‘direct’*. The second person singular written form is *‘directs’*, and the (simple) past form is *’directed’*. The semantics of the verb *(to) ‘direct’* is expressed by the property `http://dbpedia.org/on`

```
{
  "id": "108",
  "language": "EN",
  "type": "NP",
  "bindingType": "COUNTRY",
  "returnType": "CITY",
  "frameType": "NPP",
  "sentences": [
    "the capital of $x"
  ],
  "queryType": "SELECT",
  "sparqlQuery": "(bgp (triple ?subjOfProp <http://dbpedia.org/ontology/
      capital> ?objOfProp))\n",
  "sentenceToSparqlParameterMapping": {
    "$x": "subjOfProp"
  },
  "returnVariable": "objOfProp",
  "sentenceBindings": {
    "bindingVariableName": "$x",
    "bindingList": [{
    "label": "Democratic Republic of Afghanistan",
    "uri": "http://dbpedia.org/resource/Democratic_Republic_of_Afghanistan"
    }]
  },
  "combination": false
}
```

Fig. 3. 2nd grammar rule automatically generated from the entry in Figure 1

tology/director. Thus, the meaning of *'Quentin Tarantino directed Pulp Fiction'* can be expressed by the triple:

<PF> <http://dbpedia.org/ontology/director> <QT>

The entry specifies that the subject of the property is realized by the direct object of the verb *'direct'* while the object of the property is realized by the syntactic subject of the verb *'direct'*. The rule in Figure 5 is automatically generated for the above lemon entry. This rule generates the following four proto-questions: 1) *'Which person directed X?'*, 2) *'Which person directs X?'*, 3) *'Who directed X?'*, 4) *'Who directs X?'*. At the position X, either labels of individuals of type Film can be inserted or noun phrases denoting films (FILM_NP), e.g. *'films starring Bruce Willis'*, *'films produced before 1999'* etc.

2.3 Intransitive Verbs with a prepositional adjunct

As an example of an intransitive verb with a prepositional adjunct we discuss the verb *(to) 'flow through'*. The corresponding lemon entry is given in Figure 6. According to the lemon entry, the verb has a subject flow_subj and a prepositional adjunct flow_pobj. The semantics of *'X flows through Y'* is captured by the property http://dbpedia.org/ontology/city, where the subject of the property is realized by the flow_subj, and the object of the property is realized by flow_pobj. The entry states that the default domain

```
:lexicon_en a lemon:Lexicon ;
  lemon:language "en" ;
  lemon:entry :to_direct .

:to_direct a lemon:LexicalEntry ;
  lexinfo:partOfSpeech lexinfo:verb ;
  lemon:canonicalForm :form_direct ;
  lemon:otherForm :form_directs ;
  lemon:otherForm :form_directed ;
  lemon:synBehavior :direct_frame_transitive ;
  lemon:sense :direct_ontomap .

:form_direct a lemon:Form ;
  lemon:writtenRep "direct"@en ;
  lexinfo:verbFormMood lexinfo:infinitive .

:form_directs a lemon:Form ;
  lemon:writtenRep "directs"@en ;
  lexinfo:person lexinfo:thirdPerson .

:form_directed a lemon:Form ;
  lemon:writtenRep "directed"@en ;
  lexinfo:tense lexinfo:past .

:direct_frame_transitive a lexinfo:TransitiveFrame ;
  lexinfo:subject :direct_subj ;
  lexinfo:directObject :direct_obj .

:direct_ontomap a lemon:OntoMap, lemon:LexicalSense ;
  lemon:ontoMapping :direct_ontomap ;
  lemon:reference <http://dbpedia.org/ontology/director> ;
  lemon:subjOfProp :direct_obj ;
  lemon:objOfProp :direct_subj ;
  lemon:condition :direct_condition .

:direct_condition a lemon:condition ;
  lemon:propertyDomain <http://dbpedia.org/ontology/Film> ;
  lemon:propertyRange <http://dbpedia.org/ontology/Person> .
```

Fig. 4. Lemon entry for the transitive verb *(to) 'direct'*

of the property is `http://dbpedia.org/ontology/River` and the default range of the property is `http://dbpedia.org/ontology/City`.

Two grammar rules are generated from the lemon entry given in Figure 6. We do not discuss the grammar rules in detail as the rules follow the same principles as those described already. The first grammar rule is given in Figure 7 and generates sentences such as 1) *'What flows through X?'*, 2) *'What river flows through X?'*, 3) *'Which rivers flow through X?'*. The second rule is given in Figure 7 and generates the following sentences: 1) *'What does X flow through?'*, 2) *'Which cities does X flow through?'*, 3) *'Which city does X flow through?'*

2.4 Adjectives

Adjectives have a slightly different behaviour in terms of grammar rules generated compared to the entries discussed before. Let's consider the example entry for the adjective *'spanish'* in Figure 8.

The entry states that the adjective can be used in an attributive frame (e.g. *'spanish movie'*) as well as in a predicative frame (e.g. *'the movie is*

```
{
  "id": "141",
  "language": "EN",
  "type": "SENTENCE",
  "bindingType": "FILM",
  "returnType": "PERSON",
  "frameType": "VP",
  "sentences": [
    "Which person directed ($x | FILM_NP)?",
    "Which person directs ($x | FILM_NP)?",
    "Who directed ($x | FILM_NP)?",
    "Who directs ($x | FILM_NP)?"
  ],
  "queryType": "SELECT",
  "sparqlQuery": "(bgp (triple ?subjOfProp <http://dbpedia.org/ontology/director> ?
      objOfProp))\n",
  "sentenceToSparqlParameterMapping": {
    "$x": "subjOfProp"
  },
  "returnVariable": "objOfProp",
  "sentenceBindings": {
    "bindingVariableName": "$x",
    "bindingList": [{
      "label": "12 Monkeys",
      "uri": "http://dbpedia.org/resource/12_Monkeys"
    }]
  },
  "combination": false
}
```


Fig. 5. Grammar rule automatically generated from the entry in Figure 4

spanish'). The entry states that the semantics of the adjective 'spanish' can be expressed through a restriction on property `http://dbpedia.org/ontology/country` for the value `http://dbpedia.org/resource/Spain`. From this entry, two grammar rules are generated for each class in the ontology that has instances that are related to `http://dbpedia.org/resource/Spain` via the property `http://dbpedia.org/ontology/country`. In order to determine these classes, the following query is executed over the knowledge graph:

```
SELECT ?y (count(?y) as ?f) ?label WHERE {
?x <http://dbpedia.org/ontology/country>
<http://dbpedia.org/resource/Spain> ;
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
?y . ?y rdfs:label ?label . FILTER ( lang(?label)="en" ) .}
GROUP BY ?y ?label having (count(?y) > 9) order by desc(?f)
```

We thus consider only classes that have at least nine instances related to `http://dbpedia.org/resource/Spain` via the property `http://dbpedia.org/ontology/country`. The class `http://dbpedia.org/ontology/movie` is one of the classes that is returned by the above query. We generate two rules from the adjective lexical entry for *'Spanish'* for the class `http://dbpedia.org/ontology/movie`. The first rule is given in Figure 9 and generates the following full-fledged questions: 1) *'Which are Spanish movies?', 'Which is a Spanish movie?', 'Which was a Spanish movie?', 'Which were*

```
:lexicon_en a lemon:Lexicon ;
  lemon:language "en" ;
  lemon:entry :to_flow ;
  lemon:entry :through .

:to_flow a lemon:LexicalEntry ;
  lexinfo:partOfSpeech lexinfo:verb ;
  lemon:canonicalForm :form_flow ;
  lemon:otherForm :form_flows ;
  lemon:otherForm :form_flow_plural ;
  lemon:synBehavior :flow_frame ;
  lemon:sense :flow_sense1 .

:form_flow a lemon:Form ;
  lemon:writtenRep "flow"@en ;
  lexinfo:verbFormMood lexinfo:infinitive .

:form_flows a lemon:Form ;
  lemon:writtenRep "flows"@en ;
  lexinfo:number lexinfo:singular ;
  lexinfo:person lexinfo:thirdPerson ;
  lexinfo:tense lexinfo:present .

:form_flow_plural a lemon:Form ;
  lemon:writtenRep "flow"@en ;
  lexinfo:number lexinfo:plural ;
  lexinfo:person lexinfo:thirdPerson ;
  lexinfo:tense lexinfo:present .

:flow_frame a lexinfo:IntransitivePPFrame ;
  lexinfo:subject :flow_subj ;
  lexinfo:prepositionalAdjunct :flow_pobj .

:flow_sense1 a lemon:OntoMap, lemon:LexicalSense ;
  lemon:ontoMapping :flow_sense1 ;
  lemon:reference <http://dbpedia.org/ontology/city> ;
  lemon:subjOfProp :flow_subj ;
  lemon:objOfProp :flow_pobj ;
  lemon:condition :flow_condition_city .

:flow_condition_city a lemon:condition ;
  lemon:propertyDomain <http://dbpedia.org/ontology/River> ;
  lemon:propertyRange <http://dbpedia.org/ontology/City> .

:flow_obj lemon:marker :through .

:through a lemon:SynRoleMarker ;
  lemon:canonicalForm [ lemon:writtenRep "through"@en ] ;
  lexinfo:partOfSpeech lexinfo:preposition .
```

Fig. 6. Lemon entry for the intransitive verb *(to) flow through*

Spanish movies?'. A second rule not further described here would generate NP phrase such as *'a spanish movie'*. The code for our approach in addition to all relevant ressources is available here `https://github.com/fazleh2010/question-grammar-generator`. A demo of the system can be found here: `https://scdemo.techfak.uni-bielefeld.de/question-answering`.

3 Preliminary Results

We have applied our approach to the training dataset of QALD-7[4] and created a lexicon to cover the content words in these entries. This yielded a lexicon with a distribution of frame types as given in Table 1. From this lexicon, we automatically generated 5269 grammar rules using the approach described in Section 2.

[4] `https://project-hobbit.eu/challenges/qald2017/`

```
{
  "id": "536",
  "language": "EN",
  "type": "SENTENCE",
  "bindingType": "CITY",
  "returnType": "RIVER",
  "frameType": "IPP",
  "sentences": [
    "What flows through ($x | CITY_NP)?",
    "Which river flows through ($x | CITY_NP)?",
    "Which rivers flow through ($x | CITY_NP)?"
  ],
  "queryType": "SELECT",
  "sparqlQuery": "(bgp (triple ?subjOfProp <http://dbpedia.org/ontology/city> ?objOfProp))
      \n",
  "sentenceToSparqlParameterMapping": {
    "$x": "objOfProp"
  },
  "returnVariable": "subjOfProp",
  "sentenceBindings": {
    "bindingVariableName": "$x",
    "bindingList": [{
      "label": "Gaza City",
      "uri": "http://dbpedia.org/resource/Gaza_City"
    }]
  },
  "combination": false
}
```

```
{
  "id": "537",
  "language": "EN",
  "type": "SENTENCE",
  "bindingType": "RIVER",
  "returnType": "CITY",
  "frameType": "IPP",
  "sentences": [
    "What does $x flow through?",
    "Which cities does $x flow through?",
    "Which city does $x flow through?"
  ],
  "queryType": "SELECT",
  "sparqlQuery": "(bgp (triple ?subjOfProp <http://dbpedia.org/ontology/city> ?objOfProp))
      \n",
  "sentenceToSparqlParameterMapping": {
    "$x": "subjOfProp"
  },
  "returnVariable": "objOfProp",
  "sentenceBindings": {
    "bindingVariableName": "$x",
    "bindingList": [{
      "label": "Cagayan River (Mindanao)",
      "uri": "http://dbpedia.org/resource/Cagayan_River_(Mindanao)"
    }]
  },
  "combination": false
}
```

Fig. 7. 1st and 2nd grammar rules automatically generated for the entry in Figure 6

```
:lexicon_en a lemon:Lexicon ;
  lemon:language "en" ;
  lemon:entry :spanish ;
  lemon:entry :spanish_res .

:spanish a lemon:LexicalEntry ;
  lexinfo:partOfSpeech lexinfo:adjective ;
  lemon:canonicalForm :spanish_lemma ;
  lemon:synBehavior :spanish_attrFrame, :spanish_predFrame ;
  lemon:sense :spanish_sense .

:spanish_lemma lemon:writtenRep "Spanish"@en .

:spanish_predFrame a lexinfo:AdjectivePredicateFrame ;
  lexinfo:copulativeSubject :spanish_PredSynArg .

:spanish_attrFrame a lexinfo:AdjectiveAttributiveFrame ;
  lexinfo:attributiveArg :spanish_AttrSynArg .

:spanish_sense a lemon:LexicalSense ;
  lemon:reference :spanish_res ;
  lemon:isA :spanish_AttrSynArg, :spanish_PredSynArg .

:spanish_res a owl:Restriction ;
  owl:onProperty <http://dbpedia.org/ontology/country> ;
  owl:hasValue <http://dbpedia.org/resource/Spain> .
```

Fig. 8. Lemon entry for the adjective ‘spanish’

Frame type	No. of entries
NounPPFrame	59
TransitiveFrame	16
AdjectiveAttributiveFrame	5
AdjectivePPFrame	15
IntransitivePPFrame	4
Total	100

Table 1. Frequencies of entries with a certain frame type

For the questions in the training dataset of QALD-7, we manually rephrased the questions that did not follow our grammar into the closest grammar rule capturing the same meaning, if possible. 96 questions were reformulated in total, this corresponds to a percentage of 44.65% rephrased queries. While only 30.25% of questions in original form could be transformed into a SPARQL query yielding at least one answer, with the reformulated question this number raised to 54.88%. In order to use the grammar to parse NL queries, we automatically translated our grammar rules into regular expressions that parse the question and substitute the variables in the basic graph patterns by the corresponding elements matched by the regular expression. This is accomplished by looking up the values matched in the NL question with the labels in the binding lists, replacing the correct URI into the corresponding variable of the SPARQL query composed out of the basic graph patterns. Disambiguation happens in this case implicitly as the URIs selected are only those that ‘fit’ into the corresponding SPARQL tem-

```
{
  "id": "152",
  "language": "EN",
  "type": "SENTENCE",
  "bindingType": "THING",
  "returnType": "FILM",
  "frameType": "AA",
  "sentences": [
    "Which are Spanish movies?",
    "Which is a Spanish movie?",
    "Which was a Spanish movie?",
    "Which were Spanish movies?"
  ],
  "queryType": "SELECT",
  "sparqlQuery": "(bgp\n  (triple ?isA <http://dbpedia.org/ontology/country> <http://dbpedia.org/
       resource/Spain>)\n  (triple ?isA <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://
       dbpedia.org/ontology/Film>))",
  "sentenceToSparqlParameterMapping": null,
  "returnVariable": "isA",
  "sentenceBindings": {
    "bindingVariableName": "NONE",
    "bindingList": []
  },
  "combination": false
}
```


Fig. 9. Grammar rule automatically generated for the lexical entry in Figure 8

	Original questions		Rephrased questions	
	including ASK	without ASK	including ASK	without ASK
Micro Precision	58.49%	58.49%	90.41%	90.41%
Micro Recall	4.49%	4.68%	47.76%	49.85%
Micro F-measure	8.33%	8.67%	62.50%	64.26%
Macro Precision	8.93%	10.32%	41.02%	47.42%
Macro Recall	9.30%	10.75%	42.33%	48.92%
Macro F-measure	8.99%	10.39%	41.40%	47.85%

Table 2. Results for original and rephrased questions from QALD-7

plate and are mentioned in the binding list. When parsing the QALD-7 training data queries using our grammar and instantiating the corresponding SPARQL query, we get the results in Table 2. The table shows the results of our grammar-based parsing for the original questions in comparison to the rephrased questions, both for the cases including and excluding the ASK questions. We see that the results are higher for the case where the ASK questions are not included as our approach does not support them so far.
We see that with the rephrased queries we get overall a decent result of a micro F-Measure of 62.5%. Restricting the evaluation only to the SELECT queries without ASK queries, which our grammar can currently not cover, we get a micro F-measure of 64.26%.[5].

[5] The table with all rephrasings for each question can be found here: https://docs.google.com/spreadsheets/d/1Jjt_ZlDD1zbBXs3Mhdf8kIJCAuv2H9EMlJKSqyjKabU/edit#gid=2076204763

4 Related Work

Most work on question answering over linked data currently builds on systems that use machine learning to learn a model to map natural language questions into SPARQL queries (see [6] for an overview of deep learning methods applied to QALD and [1] for an overview of recent work on natural language interfaces to databases). Examples of recent systems that use machine learning techniques are systems using probabilistic graphical models [10], Bi-directional LSTMs [11], Tree-LSTMs [2] and more recently transformers [15] as well as neural machine translation models [23]. The drawback of such systems is that they can only be used for knowledge graphs or ontologies for which training data is available, limiting the applicability of such systems in the long tail.

Besides systems using machine learning techniques, there are rule-based systems that rely on a set of rules to map NL questions into a representation that can be either directly evaluated over the knowledge graph or rely on graph exploration to map the representation of the NL question into a full-fledged SPARQL query. An early system relying on rules to map dependency parse trees of questions into SPARQL queries is Aqualog [14] and its successor Poweraqua [13]. For this, Aqualog and Poweraqua rely on a number of similarity metrics to match the elements in the question to elements in the knowledge graph and/or RDF data. WDAqua [8] is a system that does not rely on syntactic analysis, but leverages the graph to induce connections between the entities and words mentioned in an NL question, constructing different hypothesis for SPARQL queries and ranking them according to a set of features. If training data is available, a learned linear combination of the features can be used for ranking. Further, there have been a number of systems that rely on pre-refined templates [20] or patterns [5] to map queries into a SPARQL query. The above mentioned systems including WDAqua, Aqualog/PowerAqua and Qakis are systems that in principle do not require training data to adapt the system to a different domain, as the interpretation of the question is guided by the underlying knowledge graph. We have proposed a different paradigm to the above mentioned systems that rely on the generation of a lexicalized question grammar from a lemon lexicon that can be used both to generate and parse questions into SPARQL queries. Our approach has the benefit that the grammars can be used in a guided QA interface, supporting a user in selecting a question that approximates his/her information need the best. Further, our approach has the advantage of supporting the portability across domains without having to provide training data, albeit requiring the creation of a lemon lexicon if not available. The basic idea of generating lexicalized grammars from ontology lexica goes back to our earlier work on generating LTAG grammars from ontology lexica [21]. While we sketched the approach in earlier work, we only worked on a very restricted domain (GEOBASE) and did not

provide the empirical proof that the approach is feasible. In this paper we have revisited the approach in the context of the lemon model and empirically shown on the QALD-7 dataset that our grammar, given the appropriate rephrasings, has a reasonable recall and precision. Our work is related to approaches to question answering over linked data building on controlled languages. The work of Ferré [9], for example, has proposed a controlled language approach to translating questions into full SPARQL 1.1, supporting all the relevant SPARQL features including joins, union, optionals, negation, quantification, aggregation/grouping, etc. However, the language proposed by Ferré, SQUALL, is not a natural language, but an artificial language using RDF elements including URIs. In contrast, our goal has been to propose an approach in which users can formulate their queries in natural language without knowing anything about the underlying data model or graph. Other natural language interfaces based on controlled natural language are based on finite-state automatons that guide a user in composing a question [16, 12]. In terms of guided interfaces to question answering over linked data, different interfaces have been proposed. In our own work, we presented two interactive guided interfaces that rely on the automatic generation of questions [4, 19]. However, the mechanisms for generating the questions were quite adhoc. In this paper we have presented a grammar formalism that is declarative and allows thus to share the grammar as an important asset. Further, the grammar supports some level of compositionality in that more complex NPs can be nested into questions. Other interactive interfaces to RDF data have been proposed as well [16, 12, 24, 3].

5 Conclusion and Future Work

In this paper we have presented an approach by which question answering grammars can be automatically generated from lemon lexica. We have proposed a specific grammar formalism that has been developed to provide a basic level of composition. We have presented preliminary results on the QALD-7 English training dataset showing that our approach is feasible and provides very good results provided that questions are rephrased with our grammar and a corresponding lemon lexicon is available. While the need to rephrase the questions can be seen as a limiting bottleneck, in the context of a guided question answering interface it is possible to guide users to formulate questions following the rules of the grammar. In this scenario our results look promising. The advantage of our approach is that, being model-based, the governance and control over the lifecycle of the QA system is more transparent, as the developer can predict the impact of adding a further lexical entry to the lemon lexicon. This is a significant advantage over machine learning based systems where the impact of adding a further example can not always be predicted. Further, many redundant examples

with the same content words might be needed for the machine learning system to learn a pattern. Our approach can be straightforwardly adapted to different ontologies and knowledge graphs for which no training datasets are available, an important advantage compared to ML-based approaches.

In the future, we will adapt our system to other languages and will develop approaches that can scale-up the acquisition of a lemon lexicon by semi-automatic means. We will also evaluate our system on other datasets including more recent versions of QALD as well as LC-QuAD 2.0.

Acknowledgements: This work has been funded by the European Commission under grant 825182 (Prêt-à-LLOD) as well as by the project 'Unbiased Bots That Build Bridges' (U3B), funded by VolkswagenStiftung.

References

1. Affolter, K., Stockinger, K., Bernstein, A.: A comparative survey of recent natural language interfaces for databases. VLDB Journal **28**, 793–819 (2019)
2. Athreya, R.G., Bansal, S., Ngomo, A.N., Usbeck, R.: Template-based question answering using recursive neural networks. CoRR **abs/2004.13843** (2020)
3. Bernstein, A., Kaufmann, E., Göhring, A., Kiefer, C.: Querying ontologies: A controlled english interface for end-users. In: Proc. of the 4th International Semantic Web Conference (ISWC). pp. 112–126 (2005)
4. Biermann, L., Walter, S., Cimiano, P.: A guided template-based question answering system over knowledge graphs. In: Proc. of the 21st International Conference on Knowledge Engineering and Knowledge Management (EKAW) (2018)
5. Cabrio, E., Cojan, J., Gandon, F., Hallili, A.: Querying multilingual dbpedia with qakis. In: Proc. of the 10th Extended Semantic Web Conference (ESWC), Satellite Events, Revised Selected Papers. vol. 7955, pp. 194–198 (2013)
6. Chakraborty, N., Lukovnikov, D., Maheshwari, G., Trivedi, P., Lehmann, J., Fischer, A.: Introduction to neural network based approaches for question answering over knowledge graphs. CoRR **abs/1907.09361** (2019)
7. Cimiano, P., Buitelaar, P., McCrae, J.P., Sintek, M.: Lexinfo: A declarative model for the lexicon-ontology interface. J. Web Semant. **9**(1), 29–51 (2011)
8. Diefenbach, D., Both, A., Singh, K., Maret, P.: Towards a question answering system over the semantic web. Semantic Web **11**(3), 421–439 (2020)
9. Ferré, S.: SQUALL: A controlled natural language as expressive as SPARQL 1.1. In: Proc. of 18th International Conference on Applications of Natural Language to Information Systems (NLDB) (2013)
10. Hakimov, S., Jebbara, S., Cimiano, P.: AMUSE: multilingual semantic parsing for question answering over linked data. In: Proc. of the 16th International Semantic Web Conference (ISWC). pp. 329–346 (2017)
11. Hakimov, S., Jebbara, S., Cimiano, P.: Evaluating architectural choices for deep learning approaches for question answering over knowledge bases. In: Proc. of the 13th IEEE International Conference on Semantic Computing (ICSC). pp. 110–113 (2019)

12. Karam, N., Streibel, O., Karjauv, A., Coskun, G., Paschke, A.: Answering controlled natural language questions over rdf clinical data. In: Proc. of the Demo/-Poster Session of the European Semantic Web Conference (ESWC) (2020)
13. López, V., Fernández, M., Motta, E., Stieler, N.: Poweraqua: Supporting users in querying and exploring the semantic web. Semantic Web **3**(3), 249–265 (2012)
14. López, V., Motta, E., Uren, V.S.: Aqualog: An ontology-driven question answering system to interface the semantic web. In: Proc. of the Human Language Technology Conference of the of the North American Chapter of the Association of Computational Linguistics (NAACL) (2006)
15. Lukovnikov, D., Fischer, A., Lehmann, J.: Pretrained transformers for simple question answering over knowledge graphs. In: Proc. of the 18th International Semantic Web Conference (ISWC). pp. 470–486 (2019)
16. Mazzeo, G.M., Zaniolo, C.: Answering controlled natural language questions on rdf knowledge bases. In: Proc. of the 19th International Conference on Extending Database Technology (EDBT) (2016)
17. McCrae, J.P., Montiel-Ponsoda, E., Cimiano, P.: Collaborative semantic editing of linked data lexica. In: Proc. of the Eighth International Conference on Language Resources and Evaluation (LREC). pp. 2619–2625 (2012)
18. McCrae, J.P., Spohr, D., Cimiano, P.: Linking lexical resources and ontologies on the semantic web with lemon. In: Proc. of the 8th Extended Semantic Web Conference (ESWC) (2011)
19. Rico, M., Unger, C., Cimiano, P.: Sorry, I only speak natural language: a pattern-based, data-driven and guided approach to mapping natural language queries to SPARQL. In: Proc. of the 4th International Workshop on Intelligent Exploration of Semantic Data (IESD) co-located with the 14th International Semantic Web Conference (ISWC) (2015)
20. Unger, C., Bühmann, L., Lehmann, J., Ngomo, A.N., Gerber, D., Cimiano, P.: Template-based question answering over RDF data. In: Proc. of the 21st World Wide Web Conference (WWW). pp. 639–648. ACM (2012)
21. Unger, C., Hieber, F., Cimiano, P.: Generating LTAG grammars from a lexicon/ontology interface. In: Proc. of the 10th International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG). pp. 61–68. Yale University (2010)
22. Usbeck, R., Ngomo, A.C.N., Haarmann, B., Krithara, A., Röder, M., Napolitano, G.: 7th open challenge on question answering over linked data (QALD-7). In: Semantic Web Evaluation Challenge. pp. 59–69. Springer International Publishing (2017)
23. Yin, X., Gromann, D., Rudolph, S.: Neural machine translating from natural language to SPARQL. CoRR **abs/1906.09302** (2019)
24. Zafar, H., Dubey, M., Lehmann, J., Demidova, E.: Iqa: Interactive query construction in semantic question answering systems. Journal of Web Semantics **64**, 100586 (2020)