Core Techniques of Question Answering Systems over Knowledge Bases: a Survey

Dennis Diefenbach, Vanessa Lopez, Kamal Singh, Pierre Maret

Published in Knowledge and Information Systems

Core Techniques of Question Answering Systems over Knowledge Bases: a Survey

Dennis Diefenbach[1], Vanessa Lopez[2], Kamal Singh[1] and Pierre Maret[1]

[1]Université de Lyon, CNRS UMR 5516 Laboratoire Hubert Curien, F-42023, Saint-Etienne, France

[2]IBM Research Ireland, Damastown Industrial Estate, Dublin

Abstract. The Semantic Web contains an enormous amount of information in the form of knowledge bases (KB). To make this information available, many question answering (QA) systems over KBs were created in the last years. Building a QA system over KBs is difficult because there are many different challenges to be solved. In order to address these challenges, QA systems generally combine techniques from natural language processing, information retrieval, machine learning and Semantic Web. The aim of this survey is to give an overview of the techniques used in current QA systems over KBs. We present the techniques used by the QA systems which were evaluated on a popular series of benchmarks: Question Answering over Linked Data (QALD). Techniques that solve the same task are first grouped together and then described. The advantages and disadvantages are discussed for each technique. This allows a direct comparison of similar techniques. Additionally, we point to techniques that are used over WebQuestions and SimpleQuestions, which are two other popular benchmarks for QA systems.

Keywords: Question Answering; QALD; WebQuestions; SimpleQuestions; Survey; Semantic Web; Knowledge base

1. Introduction

Question answering (QA) is a very old research field in computer science. The first QA systems were developed to access data over databases in the late sixties and early seventies. More recent works designed QA systems over free text, i.e., finding the part of text that contains the answer of a question from a set of documents. These are referred as information retrieval (IR) based QA systems. In the last two decades, thanks to the development of the Semantic Web, a lot of new structured data has become available on

Received 29 Jul 2016
Revised 10 Jul 2017
Accepted 30 Jul 2017

the web in the form of knowledge bases (KBs). Nowadays, there are KBs about media, publications, geography, life-science and more[1]. QA over KBs has emerged to make this information available to the end user. For a more detailed historical overview see (Lopez et al. 2011). The publication of new KBs accelerated thanks to the publication by the W3C of the de facto standards RDF[2] and SPARQL[3]. RDF is a format for publishing new KBs. SPARQL is a powerful query language for RDF, but it requires expert knowledge. The idea behind a QA system over KBs is to find in a KB the information requested by the user using natural language. This is generally addressed by translating a natural question to a SPARQL query that can be used to retrieve the desired information. In the following, when we speak about QA systems, we refer to QA systems over KBs. Due to the large number of QA systems nowadays, an in-depth analysis of all of them would require extensive work. Instead, we restrict to QA systems participating in a specific benchmark. This will allow for a comparison of the overall performance of the QA systems and gives at least a hint of how good the single techniques used in QA systems are. We concentrate here on the "Question Answering over Linked Data" (QALD) benchmark which is in line with the principles of the Semantic Web: it focuses on RDF KBs, it proposes KBs in different domains assuming that a QA system should be able to query any KB, it proposes large open-domain KB like DBpedia such that scalability becomes an issue, it offers questions where the information should be extracted from multiple interlinked KBs, it contains questions with multiple relations and operators such as comparatives and superlatives and it proposes multilingual questions assuming that the users can be of different nationalities. Moreover, we collected a list of QA systems evaluated over WebQuestions and SimpleQuestions, two popular benchmarks, and we point to approaches used by these QA systems and not by the ones evaluated over QALD.

Differently from other surveys in the field of QA over KBs, we focus on the techniques behind existing QA systems. We identified five tasks in the QA process and describe how QA systems solve them. This way each QA system participating in QALD is decomposed and compared. The tasks are question analysis, phrase mapping, disambiguation, query construction and querying distributed knowledge. Our main aim is to describe, classify and compare all techniques used by QA systems participating in QALD. This provides a good overview on how QA systems over KBs work.

The paper is organized as follows: in section 2, we position our survey with respect to past surveys on QA systems. In section 3, we describe three popular benchmarks in QA namely: WebQuestions, SimpleQuestions and QALD. In section 4, we describe how the compared QA systems were selected. In section 5, we give an overview over the five tasks. In section 6, we describe the techniques used for the question analysis task. In section 7, we discuss the techniques used for the phrase mapping task and, in section 8, the techniques for the disambiguation task are provided. In section 9, we describe how the different QA systems construct a query. Finally in section 10, we describe what changes should be made if the required information is distributed across different KBs. In section 11 and 12, we compare the analysed systems to the ones evaluated over WebQuestions and SimpleQuestions. In section 13, we give an overview of the evolution of the challenges in the field of QA. Based on the analysis of this publication, in section 14, we point to future developments in the domain of QA.

[1] http://lod-cloud.net
[2] http://www.w3.org/TR/rdf11-mt/
[3] http://www.w3.org/TR/sparql11-query/

2. Related Work

There have been various surveys on QA Systems in the field of IR driven by the Text Retrieval Conference (TREC), the Cross Language Evaluation Forum (CLEF) and NII Test Collections for IR systems (NTCIR) campaigns, see (Kolomiyets & Moens 2011) (Allam & Haggag 2012) (Dwivedi & Singh 2013).
In (Allam & Haggag 2012) (Dwivedi & Singh 2013), IR based QA systems are analysed based on three core components: (1) question analysis and classification, (2) information retrieval to extract the relevant documents with the answer and (3) answer extraction and ranking, which combine techniques from natural language processing (NLP), information retrieval (IR) and information extraction (IE), respectively (Allam & Haggag 2012). Based on the overall approach in (Dwivedi & Singh 2013), QA systems are categorised into: Linguistic, Statistical and Pattern Matching.
In these surveys, only a few domain-specific QA approaches over structured KBs are briefly described. In the open domain QA systems analysed, KBs and taxonomies are used, however they are not used to find the answer, but rather are used as features in the QA process, where answers are extracted from text. For example, they are used to support the question type classification and predict the expected answer type or to semantically enrich the parse-trees to facilitate the question-answering matching, often combined with ML approaches (Support Vector Machine, Nearest Neighbors, Bayesian classifiers, Decision Tree, etc.) or approaches that learn text patterns from text.
Differently from these surveys, we deep dive into the techniques applied by QA systems to extract answers from KBs. In particular, we focused on answering open-domain queries over Linked Data that have been proposed since the introduction of the QALD in 2011. We will see that the QALD benchmark introduces different challenges for QA systems over KBs (detailed in section 13), for which a wide variety of novel or existent techniques and their combination have been applied. Moreover, we point to some techniques used by QA systems evaluated over WebQuestions and SimpleQuestions.
The existing QALD reports (Lopez et al. 2013)(Cimiano et al. 2013)(Unger et al. 2014)(Unger et al. 2015)(Unger et al. 2016), surveys on the challenges faced by open domain QA systems over KBs (Lopez et al. 2011)(Freitas et al. 2012) and the latest survey on the topic (Höffner et al. 2016) gives short overviews for each QA system. They do not allow for a detailed comparison of the techniques used in each of the stages of the QA process. In this survey, the advantages and disadvantages of each technique applied by each of the 32 QA systems participating in any of the QALD campaigns is described.

3. Benchmarks for QA

Benchmarks for QA systems over KBs are a new development. In 2011, there was still no established benchmark that was used to compare different QA systems as described in (Lopez et al. 2011). Therefore, QA systems were tested on different KBs using small scale scenarios with ad-hoc tasks.
In the last number of years, different benchmarks for QA systems over KBs arised. The three most popular ones are WebQuestions (Berant et al. 2013), SimpleQuestion (Bordes et al. 2015) and QALD[4] (Lopez et al. 2013)(Cimiano et al. 2013)(Unger et al. 2014)(Unger et al. 2015)(Unger et al. 2016). In fact QALD is not one benchmark but

[4] http://www.sc.cit-ec.uni-bielefeld.de/qald/

Challange	Task	Dataset	Questions	Language
QALD-1	1	DBpedia 3.6	50 train / 50 test	en
	2	MusicBrainz	50 train / 50 test	en
QALD-2	1	DBpedia 3.7	100 train / 100 test	en
	2	MusicBrainz	100 train / 100 test	en
QALD-3	1	DBpedia 3.8	100 train / 99 test	en, de, es, it, fr, nl
	2	MusicBrainz	100 train / 99 test	en
QALD-4	1	DBpedia 3.9	200 train / 50 test	en, de, es, it, fr, nl, ro
	2	SIDER, Diseasome, Drugbank	25 train / 50 test	en
	3	DBpedia 3.9 with abstracts	25 train / 10 test	en
QALD-5	1	DBpedia 2014	300 train / 50 test	en, de, es, it, fr, nl, ro
	2	DBpedia 2014 with abstracts	50 train / 10 test	en
QALD-6	1	DBpedia 2015	350 train / 100 test	en, de, es, it, fr, nl, ro, fa
	2	DBpedia 2015 with abstracts	50 train / 25 test	en
	3	LinkedSpending	100 train / 50 test	en

Table 1. Overview of the QALD challenges launched so far.

a series of evaluation campaigns for QA systems over KBs. Six challenges have been proposed up till now. An overview is given in Table 1. Each year one of the tasks included QA over DBpedia. In the following, we concentrate on these results. Moreover, to address the case where the requested information is distributed across many KBs, we consider QALD-4 Task 2. It offers 3 interlinked KBs, 25 training questions and 25 test questions that can be answered only by searching and combining information distributed across the 3 KBs. We compare these three benchmarks based on the dataset, the questions and the evaluation metrics.

3.1. Datasets

WebQuestions and SimpleQuestions contain questions that can be answered using Freebase(Google 2016). The QALD challenges always included DBpedia[5] as an RDF dataset besides some others. The main difference between DBpedia and Freebase is how the information is structured. Freebase contains binary and also non-binary relationships. Non-binary relationships are stored using one type of reification that uses "mediator nodes". An example is given in Figure 1. This is not the case in DBpedia where only binary relations are stored. In particular temporal information is not present. Some of the works over WebQuestions like (Bordes et al. 2015) and (Jain 2016) converted non-binary relationships to binary relationships by dropping the "mediator node". (Bordes et al. 2015) noted that this way still 86% of the questions in WebQuestions can be answered. Moreover, for each relationship, Freebase also stores its inverse, this is not the case for DBpedia since they are considered redundant. Regarding the size, the DBpedia endpoint contains nearly 400.000 triples while the last Freebase dump contains 1.9 billion triples. Note that due to reification and due to the materialization of the inverse of each property, this is roughly the same amount of information. Moreover note that WebQuestions and SimpleQuestions are often tackeled by using only a subset of Freebase. This is denoted as FB2M and FB5M containing only 2 and 5 million entities, respectively.

[5] http://www.dbpedia.org/

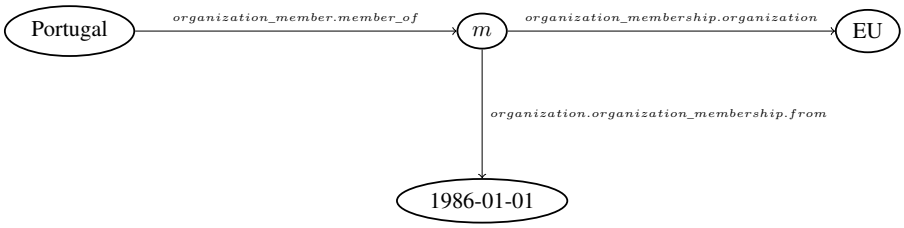

Figure 1. Freebase also contains non-binary relationships like "Portugal is a member of the European Union since 1986.". This information is stored in Freebase using one type of reification. It uses "mediator nodes" (in this example m). Note that in the dump, for each relationship, its inverse is also stored. This is not the case with DBpedia.

3.2. Questions

WebQuestions contains 5810 questions which follow a very similar pattern due to the crowd-sourcing method used to construct the dataset. Around 97% of them can be answered using only one reified statement with potentially some constraints like type constraints or temporal constraints[6]. SimpleQuestion contains 108.442 questions that can be answered using one binary-relation.
The questions of QALD are prepared each year by the organizers of the challenge. They can be generally answered using up to three binary-relations and often need modifiers like ORDER BY and COUNT. Moreover, some of the questions are out of scope. The training set is rather small with about 50 to 250 questions. This results in few opportunities for supervised learning. Finally, while the SimpleQuestion and the QALD datasets are annotated with a SPARQL query for each question, the WebQuestions dataset is only annotated with the labels of the answers. There exists a dataset containing SPARQL queries for each of the question in WebQuestions, which is called WebQuestionsSP (Yih et al. 2016).

3.3. Evaluation

For evaluation, three parameters are used: precision, recall, and F-measure. They are the same across all benchmarks. Precision indicates how many of the answers are correct. For a question q it is computed as:

$$\text{Precision(q)} = \frac{\text{number of correct system answers for q}}{\text{number of system answers for q}}.$$

The second parameter is recall. For each question there is an expected set of correct answers, these are called the gold standard answers. Recall indicates how many of the returned answers are in the gold standard. It is computed as:

$$\text{Recall(q)} = \frac{\text{number of correct system answers for q}}{\text{number of gold standard answers for q}}.$$

We explain these parameters using the following example. Assume a user asks the question: "Which are the three primary colors?". The gold standard answers would be "green", "red" and "blue". If a system returns "green" and "blue", as the answers, then

[6] https://www.microsoft.com/en-us/download/details.aspx?id=52763

the precision is 1, since all answers are correct but the recall is only 2/3 since the answer "red" is missing.
The global precision and recall of a system can be computed into two ways, they are denoted as the micro and macro precision and recall, respectively. The micro precision and recall of a system are computed by taking the average of the precision and recall over all answered questions, i.e. non answered questions are not taken into account. The macro precision and recall are computed by taking the average of the precision and recall over all questions. The (micro or macro) F-measure is the weighted average between the (micro or macro) precision and recall and it is computed as follows:

$$\text{F-measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

Note that the F-measure is near to zero if either precision or recall are near to zero and is equal to one only if both precision and recall are one. In this survey we report macro precision, recall and F-measure since these are the official metrics used by the benchmarks. Another parameter which is important for the user, is runtime. This is generally indicated as the average time taken by the QA system for answering the questions.

4. Selection of the QA systems

We briefly describe how the QA systems compared in this paper were selected. We consider the QA systems that either directly participated in the QALD challenges or that were evaluated afterwards reusing the same evaluation set-up. To identify the latter systems, we searched in Google Scholar for all publications mentioning or citing the publications of the QALD challenges (Lopez et al. 2013)(Cimiano et al. 2013)(Unger et al. 2014)(Unger et al. 2015)(Unger et al. 2016). From among these, we took the publications referring to QA systems.
We excluded systems that were only able to answer questions formulated in a controlled natural language namely squall2sparql (Ferré 2013) and CANaLI (Mazzeo & Zaniolo 2016). Moreover, we exclude systems that propose a graphical user interface to guide users through the construction of a query. These include Sparklis (Ferré 2017) and Scalewelis (Joris & Ferré 2013).
Note that over the last years other types of QA systems have also emerged. Between these there is hybrid QA, i.e. QA using both free text and KBs. This problem was tackled by the QA systems HAWK (Usbeck et al. 2015) and YodaQA (Baudiš & Šedivỳ 2015). Another type of QA system is QA over statistical data expressed using the RDF Data Cube vocabulary. This problem was tackled by CubeQA (Höffner & Lehmann 2015) and QA^3 (Atzori et al. 2016). Since there are only a few works in these directions, we direct the readers directly to them.
The QA systems APAQ, AskDBpedia, KWGAnswer, MHE, NbFramework, PersianQA, RO_FII and UIQA that participated in QALD were excluded since no corresponding publication could be found.
Table 2 contains a list of the considered QA system with the scores they achieved on the QALD test sets.

5. The question answering process

To classify the techniques used in QA systems we divide the QA process into five tasks: question analysis, phrase mapping, disambiguation, query construction and querying

QA system	Lang	Total	Precision	Recall	F-measure	Runtime	Reference
QALD-1							
FREyA (Damljanovic et al. 2010)	en	50	0.54	0.46	0.50	36 s	(Lopez et al. 2013)
PowerAqua (Lopez et al. 2012)	en	50	0.48	0.44	0.46	20 s	(Lopez et al. 2013)
TBSL (Unger et al. 2012) ∗	en	50	0.41	0.42	0.42	-	(Unger et al. 2012)
Treo (Freitas & Curry 2014) ∗	en	50	-	-	-	-	-
QALD-2							
SemSeK (Aggarwal & Buitelaar 2012)	en	100	0.35	0.38	0.37	-	(Lopez et al. 2013)
BELA (Walter et al. 2012) ∗	en	100	0.19	0.22	0.21	-	(Walter et al. 2012)
QAKiS (Cabrio et al. n.d.)	en	100	0.14	0.13	0.13	-	(Lopez et al. 2013)
QALD-3							
gAnswer (Zou et al. 2014) ∗	en	100	0.40	0.40	0.40	≈ 1 s	(Zou et al. 2014)
RTV (Giannone et al. 2013)	en	99	0.32	0.34	0.33	-	(Cimiano et al. 2013)
Intui2 (Dima 2013)	en	99	0.32	0.32	0.32	-	(Cimiano et al. 2013)
SINA (Shekarpour et al. 2015) ∗	en	100	0.32	0.32	0.32	≈ 10-20 s	(Shekarpour et al. 2015)
DEANNA (Yahya et al. 2012) ∗	en	100	0.21	0.21	0.21	≈ 1-50 s	(Zou et al. 2014)
SWIP (Pradel et al. 2012)	en	99	0.16	0.17	0.17	-	(Cimiano et al. 2013)
Zhu et al. (Zhu et al. 2015) ∗	en	99	0.38	0.42	0.38	-	(Zhu et al. 2015)
QALD-4							
Xser (Xu et al. 2014)	en	50	0.72	0.71	0.72	-	(Unger et al. 2014)
gAnswer (Zou et al. 2014)	en	50	0.37	0.37	0.37	0.973 s	(Unger et al. 2014)
CASIA (He et al. 2014)	en	50	0.32	0.40	0.36	-	(Unger et al. 2014)
Intui3 (Dima 2014)	en	50	0.23	0.25	0.24	-	(Unger et al. 2014)
ISOFT (Park et al. 2014)	en	50	0.21	0.26	0.23	-	(Unger et al. 2014)
Hakimov et al. (Hakimov et al. 2015) ∗	en	50	0.52	0.13	0.21	-	(Hakimov et al. 2015)
QALD-4 Task 2							
GFMed (Marginean 2017)	en	25	0.99	1.0	0.99	-	(Unger et al. 2014)
SINA (Shekarpour et al. 2015) ∗†	en	25	0.95	0.90	0.92	≈ 4-120 s	(Shekarpour et al. 2015)
POMELO (Hamon et al. 2014)	en	25	0.87	0.82	0.85	≈ 2 s	(Unger et al. 2014)
Zhang et al. (Zhange et al. 2016) ∗	en	25	0.89	0.88	0.88	-	(Zhange et al. 2016)
TR Discover (Song et al. 2015) ∗	en	25	0.34	0.80	0.48	-	(Song et al. 2015)
QALD-5							
Xser (Xu et al. 2014)	en	50	0.74	0.72	0.73	-	(Unger et al. 2015)
QAnswer (Ruseti et al. 2015)	en	50	0.46	0.35	0.40	-	(Unger et al. 2015)
SemGraphQA (Beaumont et al. 2015)	en	50	0.31	0.32	0.31	-	(Unger et al. 2015)
YodaQA (Baudiš & Sedivỳ 2015)	en	50	0.28	0.25	0.26	-	(Unger et al. 2015)
QALD-6							
UTQA (Pouran-ebn veyseh 2016)	en	100	0.82	0.69	0.75	-	(Unger et al. 2016)
UTQA (Pouran-ebn veyseh 2016)	es	100	0.76	0.62	0.68	-	(Unger et al. 2016)
UTQA (Pouran-ebn veyseh 2016)	fs	100	0.70	0.61	0.65	-	(Unger et al. 2016)
SemGraphQA (Beaumont et al. 2015)	en	100	0.70	0.25	0.37	-	(Unger et al. 2016)

† Evaluated on the training set

Table 2. This table summarizes the results obtained by the QA systems evaluated over QALD. We included the QALD tasks querying DBpedia and QALD-4 task 2 which addresses the problem of QA over interlinked KBs. We indicated with "∗" the systems that did not participate directly in the challenges, but were evaluated on the same benchmark afterwards. We indicate the average running time of a query for the systems where we found it. Even if the runtime evaluations were executed on different hardware, it still helps to give an idea about the scalability.

distributed knowledge (see Figure 2).
In the following, we briefly describe the first four steps using a simple example. Assume the user inputs the question:

"What is the population of Europe?"

and that DBpedia is the queried KB.
In the question analysis task, we include all techniques that use purely syntactic features to extract information about a given question. These include, for example, determining the type of the question, it is a "What" question, identifying named entities, like "Europe", identifying relations and entities (subjects and objects), like the relation "is the population of", and their dependencies, like that there is a dependency between "is the population of" and "Europe".
In the phrase mapping task, one starts with a phrase s and tries to identify resources that with high probability correspond to s. For example, for the phrase "Europe" possible

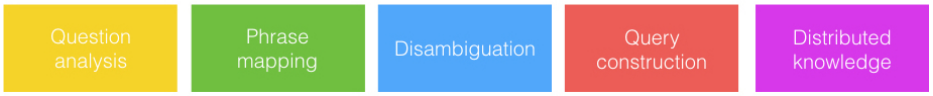

Figure 2. Tasks in the QA process

resources in DBpedia are: dbr:Europe _(band)[7] (that refers to a Band called Europe), dbr: Europe (that refers to Europe as a continent) and dbr:Europe _(dinghy) (a particular type of boat).
In the disambiguation task, we include techniques that are used to determine which of the resources identified during the phrase mapping task are the right ones. In the above example, "Europe" cannot refer to a band or a boat since it does not make sense to speak about their population. Therefore, these two resources can be excluded.
The query construction task describes how to construct a SPARQL query that can be send to an endpoint. This part also covers the construction of queries that require special operators such as comparatives and superlatives. In the example above, the generated SPARQL query would be:

```
Select ?p where {
    dbr:Europe dbp:populationTotal ?p
}
```

In the distributed knowledge task, we include techniques that are used in the case when the information to answer the question must be retrieved from several KBs. Note that QA systems generally do not follow such a strict division in five tasks. However, based on the current state of the art (including systems not reviewed here because they were not evaluated for QALD), we have found that QA systems can typically be decomposed in this way. Note that the subdivision in tasks does not aim to present a general architecture for QA systems, but to classify and compare QA systems. Moreover, in general there is no preferred order to address the tasks, i.e. Xser (Xu et al. 2014) first performs disambiguation and then constructs the query while SEMPRE (Berant et al. 2013) proceeds the other way around.
In section 10, the tasks are mapped to the challenges identified in the state of the art for QA systems. The following five sections, describe the techniques that are used by QA systems to solve the different tasks. We focus on techniques that can be adapted to any KB, i.e. techniques that do not use some DBpedia/Wikipedia specific features.

6. Question analysis

In this step, the question of the user is analyzed based on purely syntactic features. QA systems use syntactic features to deduce, for example, the right segmentation of the question, determine which phrase corresponds to an instance (subject or object), property or class and the dependency between the different phrases. Note that some systems like Xser (Xu et al. 2014) decide already in this step what part of the question corresponds to an instance, relation and class. Other systems like PowerAqua (Lopez et al. 2012) use this step to find how the terms in the question relate to each other and make an hypothesis about the correspondence. The right one can be selected afterwards. We classify techniques for question analysis in techniques for recognizing named entities,

[7] @prefix dbr: <http://dbpedia.org/resource/>

WRB	VBD	DT	NNP	NNP	VBN	.
When	was	the	European	Union	founded	?

Figure 3. POS tags returned by the Stanford POS tagger http://nlp.stanford.edu/software/tagger.shtml for the question "When was the European Union founded?". For example DT stands for determiner, NNP for proper singular noun.

techniques for segmenting the question using Part-of-speech (POS) tagging and techniques to identify dependencies using parsers.

6.1. Recognizing named entities

An important task in QA is to segment the question, i.e. identify contiguous spans of tokens that refer to a resource. As an example consider the question: "Who directed One Day in Europe?". The span of tokens "One Day in Europe" refers to a film. QA systems use different strategies to solve this problem.

Named Entity Recognition (NER) tools from NLP
One approach is to use NER tools used in natural language processing. Unfortunately, NER generally are adapted to a specific domain. It was observed in (He et al. 2014) that the Stanford NER tool[8] could recognize only 51.5% of the named entities in the QALD-3 training set.

N-gram strategy
A common strategy is to try to map n-grams (groups of n words) in the question to entities in the underlying KB. If at least one resource is found, then the corresponding n-gram is considered as a possible named entity (NE). This is used for example in SINA (Shekarpour et al. 2015) and CASIA (He et al. 2014). This has the advantage that each NE in the KB can be recognized. The disadvantage is that many possible candidates will appear and that the disambiguation will become computationally expensive.

Entity Linking (EL) Tools
Some engines use other tools to recognize NE. These are DBpedia Spotlight (Daiber et al. 2013) and AIDA (Yosef et al. 2011). These tools do not only recognize NE, but also find the corresponding resources in the underlying KB, i.e. they perform entity linking. As a result, they perform many steps at once: identifying contiguous span of tokens that refer to some entity, find possible resources that can correspond to it and disambiguate between them.

6.2. Segmenting the question using POS tagging

POS tags are used mainly to identify which phrases correspond to instances (subjects or objects), to properties, to classes and which phrases do not contain relevant information. An example of a POS tagged question is given in Figure 3. Often nouns refer to instances and classes (like "European") and verbs to properties (like "founded" above). This does not always hold. For example in the question "Who is the director of Star

[8] http://nlp.stanford.edu/software/CRF-NER.shtml

none	V-B	C-B	none	none	E-B	E-I	R-B	.
By	which	countries	was	the	European	Union	founded	?

Figure 4. Question annotated using the CoNLL IOB format. E-B indicates that the entity is beginning and E-I that it is continuing (this way phrases with more words are labeled).

Wars?" the noun "director" may refer to a property (e.g., `dbo:director`).
The general strategy using POS tags is to identify some reliable POS tags expressions to recognize entities, relations and classes. These expressions can then be easily identified using regular expressions over the POS tags.

Handmade rules
Some engines rely on hand written regular expressions. This is the case for PowerAqua ((Lopez et al. 2012)(Lopez et al. 2007)), Treo ((Freitas & Curry 2014)) and DEANNA ((Yahya et al. 2012)). PowerAqua uses regular expressions, based on the GATE NLP tool ((Cunningham et al. 2002)), to identify the question type and to group the identified expressions into triples of phrases. To do this PowerAqua relies on an extensive list of question templates, i.e., depending on the type of the question and its structure, the phrases are mapped to triples.

Learning rules
Instead of writing handmade rules, it was proposed to use machine learning algorithms to detect them (Xu et al. 2014). The idea is to create a training set where questions are tagged with entities (E), relations (R), classes (C), variables (V) and "none" tags using the CoNLL IOB format (inside-outside-beginning) as in Figure 4. Once a training corpus is constructed it is used to build a phrase tagger. The features used by the phrase tagger built in Xser ((Xu et al. 2014)) are: POS-tags, NER-tags and the words of the question itself. This way one can construct a tagger that is able to identify entities, relations, classes and variables in a given question and learn automatically rules to do that starting from the training data. A similar approach was followed also by UTQA ((Pouran-ebn veyseh 2016)). The disadvantage is that a training corpus must be created. The advantage is that no handmade rules have to be found.

6.3. Identifying dependencies using parsers

POS tagging information does not allow to identify the relation between the different chunks in a question. This is the reason for using parsers.
A parser is based on a formal grammar. A formal grammar consists of symbols (words) and production rules that are used to combine the symbols. Given a sentence, a parser computes the combination of production rules that generate the sentence according to the underlying grammar. Examples will be given in the next section. Note that there are different types of formal grammars. The systems participating to the QALD challenges use parsers based on different types of grammars. We present them below and show their advantages.

6.3.1. Parsers based on phrase structure grammars

Some parsers rely on phrase structure grammars. The idea of phrase structure grammars is to break down a sentence into its constituent parts. An example is given in figure 5.

These types of trees are used similarly to POS tags, i.e. one tries to find some graph

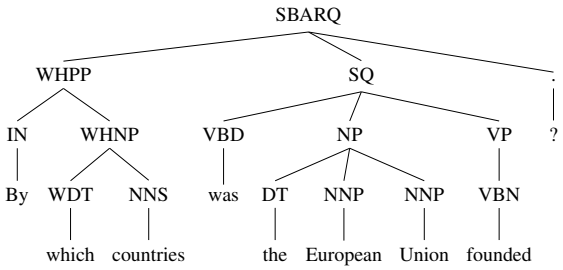

Figure 5. Parsing tree of the question "By which countries was the European Union founded?" returned by the Stanford Parser (http://nlp.stanford.edu/software/lex-parser.shtml). At the bottom of the tree are the words in the question and the corresponding POS tags. The tags above denote phrasal categories like noun phrase (NP).

patterns that map to instances, properties or classes with high confidence. Differently from POS tags, one can deduce the dependencies between entities, relations and classes. Parsers of such type are used in the QA systems Intui2 ((Dima 2013)), Intui3 ((Dima 2014)) and Freya ((Damljanovic et al. 2010)).

6.3.2. Parsers based on dependency grammars

The idea behind dependency grammars is that the words in a sentence depend on each other, i.e. a word "A" depends on a word "B". "B" is called the head (or governor) and "A" is called the dependent. Moreover, parsers also generally indicate the type of relation between "A" and "B".

Standford dependencies, Universal dependencies
Figure 6 shows the result of the Stanford dependency parser for the questions "By which countries was the European Union founded?".

For example, the tree indicates that "founded" is the head of "By", or that "By" is the dependent of "founded". The Stanford parser also returns the type of the dependency, in the example above "prep" indicates that "by" is a preposition of "founded". The advantages of using dependency trees can be seen by looking at the example above. In the original question, the words in the relational expression "founded by" are not subsequent which makes them difficult to extract. This is not the case in the dependency tree where "by" is connected to "founded". This is the main reason why dependency representations are used for relation extraction. Moreover, the parsing tree contains grammatical relations like nsubj (nominal subject), nsubjpass (nominal passive subject) and others which can be used to identify the relations between different phrases in the question.
A method to extract relations is to search the dependency tree for the biggest connected subtree that can be mapped to a property. In the example above this would be the subtree consisting of the nodes "founded" and "by". The arguments associated with a relation are then extracted searching around the subtree. This is for example used by gAnswer (Zou et al. 2014). Another strategy is to first identify named entities in the dependency tree and choose the shortest path between them as a relation. In the example above these are the entities "country" and "European Union". This is used in the pattern library PATTY (Nakashole et al. 2012).

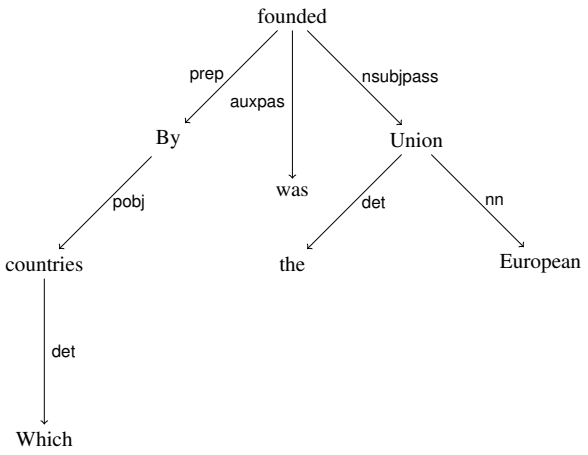

Figure 6. Result of the Stanford dependency parser for the questions "By which countries was the European Union founded?".

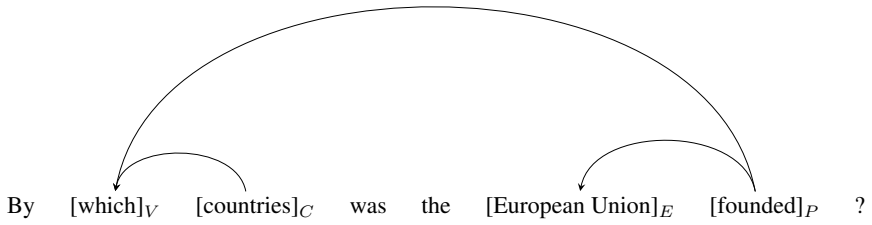

Figure 7. Parse tree for the questions "By which countries was the European Union founded?" using the Stanford dependency parser

Phrase dependencies and DAGs

While the Stanford dependency parser considers dependencies between words, the system Xser (Xu et al. 2014) considers dependencies on a phrase level, namely between phrases referring to entities, relations, classes and variables as in 6.2. The following dependency relation is considered: relations are the head of the two corresponding arguments and classes are the head of the one corresponding argument (arguments are either entities, variables or other relations and classes). As an example, look at the dependency graph of the question "By which countries was the European Union founded?" in figure 7. The phrase "founded" is the head of the phrase "which" and the entity "European Union". The class "country" is the head of the variable "which". Note that this is not a tree, but a direct acyclic graph (DAG). To compute the DAG, Xser uses a SHIFT-REDUCED parser that is trained on a manually annotated dataset.

The parser uses as features the POS tags of the words, the type of the phrase and the phrase itself. The advantage is that the parser learns automatically which is the relation between the phrases. The disadvantage is that one needs a manually annotated corpus.

	NER	NE n-gram startegy	EL tools	POS hand-made	POS learned	Parser structural grammar	Dependency parser	Phrase dependencies and DAG	Reference
BELA				x					(Walter et al. 2012)
CASIA		x			x		x		(He et al. 2014)
DEANNA		x		x			x		(Yahya et al. 2012)
FREyA						x			(Damljanovic et al. 2010)
gAnswer							x		(Zou et al. 2014)
GFMed									(Marginean 2017)
Hakimov et al.								x	(Hakimov et al. 2015)
Intui2						x			(Dima 2013)
Intui3	x			x					(Dima 2014)
ISOFT			x	x			x		(Park et al. 2014)
POMELO		x							(Hamon et al. 2014)
PowerAqua				x					(Lopez et al. 2012)
QAKiS	x	x							(Cabrio et al. n.d.)
QAnswer		x					x		(Ruseti et al. 2015)
RTV	?						x		(Giannone et al. 2013)
SemGraphQA		x					x		(Beaumont et al. 2015)
SemSeK		x				x	x		(Aggarwal & Buitelaar 2012)
SINA		x							(Shekarpour et al. 2015)
SWIP	?						x		(Pradel et al. 2012)
TBSL				x					(Unger et al. 2012)
TR Discover									(Song et al. 2015)
Treo				x			x		(Freitas & Curry 2014)
UTQA					x				(Pouran-ebn veyseh 2016)
Xser	?				x			x	(Xu et al. 2014)
Zhang et al.		x							(Zhange et al. 2016)
Zhu et al.						x			(Zhu et al. 2015)

Table 3. Each line of this table corresponds to a QA system. The columns indicate which of the techniques presented in the question analysis task (left table) and in the phrase mapping task (right table) is used. The columns refer to the subsections of section 6 and 7. We put a question mark if it is not clear from the publication which technique was used.

6.4. Summary and research challenges for question analysis

Table 3 shows which strategy is used by the different engines for question analysis. We put a question mark if it is not clear from the description of the publication which technique is used. The research challenges that have been identified are the following: the identification of question types, the multilinguality and the identification of aggregation, comparison and negation operators. Section 13 will give a transversal view on research challenges through the surveys where they are discussed.

7. Phrase mapping

This step starts with a phrase (one or more words) *s* and tries to find, in the underlying KB, a set of resources which correspond to *s* with high probability. Note that *s* could correspond to an instance, a property, or a class. As an example, the phrase *EU* could correspond to the DBpedia resources `dbr:European_Union` but also to `dbr:University_of_Edinburgh` or `dbr:Execution_unit` (a part of a CPU). In the following, we want to provide an overview of the techniques used by QA systems to deal with these problems.

7.1. Knowledge base labels

RDF Schema introduces the property `rdfs:label`[9] to provide a human-readable version of a resource's name. Moreover, multilingual labels are supported using language tagging. For example, the following triples appear in DBpedia:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>;
PREFIX dbr: <http://dbpedia.org/resource/>;
dbr:European_Union rdfs:label "European␣Union"@en .
dbr:European_Union rdfs:label "Europäische␣Union"@de .
dbr:European_Union rdfs:label "Union␣européenne"@fr .
```

saying that the human-readable version of `dbr:European_Union` in french is `Union européenne`. By using the RDF Schema convention the resources corresponding to a phrase s can be found by searching all resources r in the KB whose label $label(r)$ is equal to or contains s. This strategy is used by all QA systems.
To answer a question in reasonable time some type of index is needed to search through all the labels. There are two common choices. The first is to use a triple-store build-in index like in Virtuoso[10]. This is for example used by SINA (Shekarpour et al. 2015). The second is to use Lucene. Engines that use Lucene are PowerAqua (Lopez et al. 2012) and SemSeK (Aggarwal & Buitelaar 2012). In both cases one can search very fast through the labels. The advantage of a build-in triple-store index is that it is kept synchronized with the corresponding KB.
Several problems can arise that can be grouped in two categories. The first category contains problems that arise because the phrase s, as a string, is similar to $label(r)$ but not a strict subset. For example, s could be misspelled, the order of the words in s and $label(r)$ could be different or singular plural variations (like city/cites).
The second category contains problems that arise because s and $label(r)$ are completely different as strings but similar from a semantic point of view. This can happen for example when the questions contains abbreviations like *EU* for `dbr:European_Union`, nicknames like *Mutti* for `dbr:Angela_Merkel` or relational phrases like *is married to* corresponding to the DBpedia property `dbo:spouse`. All these problems arise because the vocabulary used in the question is different from the vocabulary used in the KB. The term "lexical gap" is used to refer to these problems.

7.2. Dealing with string similarity

One possibility to deal with misspelling is to measure the distance between the phrase s and the labels of the different resources of the KB. There are a lot of different distance metrics that can be used. The most common one are the Levenshtein distance and the Jaccard distance. Lucene for example offers fuzzy searches which return words similar to the given one based on the Levenshtein Distance or another edit distance. Moreover Lucene offers also stemming which allows to map "cities" to "city" since the stem is in both cases "citi".

[9] @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
[10] `https://github.com/openlink/virtuoso-opensource`

7.3. Dealing with semantic similarity

In this section we want to consider the phrase mapping problem when s and $label(r)$ have only a semantic relation.

7.3.1. Databases with lexicalizations

One frequent approach is to use a lexical database. Examples for such databases are WordNet and Wiktionary. The strategy is to first expand s by synonyms $s_1, ..., s_n$ found in the lexical database. Then $s_1, ..., s_n$ are used to retrieve the resources instead of s. For example, WordNet returns for *EU* the synonyms *European Union, European Community, EC, European Economic Community, EEC, Common Market, Europe* which are then used to retrieve the corresponding resources in the KB. More precisely WordNet groups words in sets of roughly synonymous words called synsets. For example, the word *EU* is contained in two synsets. The first was described above and refers to *"an international organization of European countries formed after World War II to reduce trade barriers and increase cooperation among its members"*. The second synset contains *europium, Eu, atomic number 63* and refers to *"a bivalent and trivalent metallic element of the rare earth group"*.
WordNet is used for example in TBSL (Unger et al. 2012), PowerAqua (Lopez et al. 2012) and SemSek (Aggarwal & Buitelaar 2012). The main advantage is that more mappings can be resolved improving recall. The disadvantage is that the expansion of s increases the number of possible corresponding resources. This can affect precision and has the side effect that the disambiguation process becomes computationally more heavy. Moreover lexical databases are often domain-independent and cannot help for domain-specific mappings.

7.3.2. Redirects

Another way to collect new labels of a resource is to follow, if present, the owl:sameAs links. The labels in the connected KB can be used then in the same way as in the original KB. Moreover, the anchor texts of links, mapping to an KB resource, can also be used. The labels generated by the anchor texts of the Wikipedia links are contained in DBpedia lexicalizations[11].

7.3.3. A database with relational lexcalizations (PATTY)

PATTY (Nakashole et al. 2012) is a Database with relational lexicalizations. The main objective is to collect natural language expressions and group them if they refer to the same relation. These groups are called "pattern synsets" in analogy of Wordnet. Moreover, the pattern synsets are organized in a taxonomy like synsets in WordNet. An example of such a pattern synset is *(is album, [[num]] album by, was album, [[det]] album with, [[adj]] album by)* where *[[num]]* corresponds to a cardinal number, *[[det]]* corresponds to a determiner and *[[adj]]* corresponds to an adjective (for this reason they are called "patterns"). Examples where different phrases express the same relation are: "Beatles*'s album* Revolver ...." , "Revolver is a *1966 album by* the Beatles" , "Revolver is *an album with* the Beatles", and "Revolver is the *seventh album by* the Beatles".
PATTY can be used as standard database with lexicalizations as in 7.3.1 and shares their advantages and disadvantages. PATTY is for example used by Xser (Xu et al. 2014).

[11] http://wiki.dbpedia.org/lexicalizations

7.3.4. Finding mappings using extracted knowledge

There are different tools that extract binary relations expressed in natural language from text corpora. An example of such a binary relation is *("Angela Merkel", "is married to", "Joachim Sauer")*. Note that it is a relation expressed in natural language so the subject, object and predicate are not mapped to any KB. Tools that can extract this type of triples are for example ReVerb (Fader et al. 2011), TEXTRUNNER (Yates et al. 2007), WOE (Wu & Weld 2010) and PATTY (Nakashole et al. 2012). An approach that use this data to find natural language representations of properties in a KB was described in (Berant et al. 2013). It assumes that the subject and object of the binary relation are mapped to instances in an underlying KB. First the relational phrase is normalized to an expression rel. In a second step the classes of the subject and object are added obtaining $rel[class1, class2]$, i.e. in the example above *married[Person, Person]*. Then all entity pairs that are connected in the text by the relation rel and that matches the classes are computed. Denote this set as $Sup(rel[class1, class2])$. Moreover, for all properties in the KB the set of connected entity pairs are computed. For a property p we denote this set as $Sup(p)$. The relation $rel[class1, class2]$ is then aligned to the property p if the domain and range of both agree and $Sup(rel[class1, class2]) \cap Sup(p) \neq \emptyset$. This technique was used by CASIA (He et al. 2014). The main disadvantage here is that the data can be very noisy. A similar approach is described in gAnswer (Zou et al. 2014).

7.3.5. Finding mappings using large texts

There are two methods that use large texts to find natural language representation for phrases. The first is the core of the BOA framework (Gerber & Ngomo 2011) and M-Atoll (Walter et al. 2014). Both take as an input a property p contained in a KB and try to extract natural language expressions for p from a large text corpus. To do that both extract from the KB the subject-object pairs *(x,y)* that are connected by the property p. Then the text corpus is scanned and all sentences are retrieved that contain both $label(x)$ and $label(y)$). At the end the segments of text between $label(x)$ and $label(y)$, or $label(y)$ and $label(x)$ are extracted. The idea is that these text segments are a natural language representation of the property p. These text segments are stored together with the range and domain of p and ranked such that the top ranked patterns are a more likely natural language representation of p. Using this data, a relational phrase r can be mapped to a property p by searching a similar text fragment in the BOA framework. This technique is for example used in TBSL (Unger et al. 2012). Moreover Qakis (Cabrio et al. n.d.) uses the pattern library WikiFrameworks (Mahendra et al. 2011) that was constructed in a very similar way. Also QAnswer (Ruseti et al. 2015) uses this approach, but the relational phrase is extracted using dependency trees. The big advantage of this approach is that the extracted relational phrases are directly mapped to the corresponding properties. The relational expressions found are tightly connected to the KB, differently to PATTY where the relations are more KB-independent.
Another approach that uses large texts is based on distributional semantics. The idea behind distributional semantics is that if two words are similar then they appear in the same context. An n-dimensional vector v_i is associated with each word w_i. The vectors are created such that words that appear in the same context have similar vectors. From among the QA systems participating in QALD, two tools that are based on distributional semantics are: word2vec[12] and Explicit Semantic Analysis (ESA)[13]. In both cases the

[12] https://code.google.com/p/word2vec/
[13] http://code.google.com/p/dkpro-similarity-asl/

vectors that are associated with the words have the property that the cosine similarity of a given pair of words is small if the words are similar. In the case of word2vec, the experimental results showed that the closest vectors to the vector of France *vec(France)* are the vectors *vec(Spain), vec(Belgium), vec(Netherlands), vec(Italy)*. Moreover, the vector *vec(queen)* is very near to the vector *vec(king)-vec(man)+vec(woman)*. In this sense, the semantics of the words are reflected into their associated vectors. The general strategy in the phrase mapping task is the following. Let us assume that there is a phrase *s* and a set of possible candidates $\{x_1, ..., x_n\}$ which can be instances, relations or classes. Then the vector representation v_0 of *s* and $v_1, ..., v_n$ of the lexicalizations of $x_1, ..., x_n$ are retrieved. Since the similarity of the words is reflected in the similarity of the vectors, the best candidates from $\{x_1, ..., x_n\}$ are the ones whose vectors are more similar to v_0. For example if the right semantics are captured then the vector $vec(spouse)$ should be similar to the vector of $vec(married)$. The main advantage is that this technique helps to close the lexical gap. However, the disadvantages are that it can introduce noise and that it is generally a quite expensive operation. For this reason the possible candidate set is generally not the entire set of instances, relations and classes, but only a subset of them. CASIA (He et al. 2014) for example uses this technique only for classes and uses word2vec as the tool. Treo uses a strategy such that it can assume that a phrase *s* corresponds to a property or class of a particular instance. In this case the candidate set contains only 10-100 elements. Here ESA is used as the tool.

7.4. Wikipedia specific approaches

Some engines use other tools for the phrase mapping task namely: DBpedia lookup[14] and the Wikipedia Miner Tool[15]. The first is for example used by gAnswer (Zou et al. 2014) the second by Xser (Xu et al. 2014) and Zhu et al. (Zhu et al. 2015).

7.5. Summary and research challenges for the phrase mapping task

Table 4 gives an overview showing which technique is used by which engine for phrase mapping. The important point in this step is that one has to find a balance between selecting as few candidate resources as possible to improve precision and time performance, and select enough candidates so that the relevant one is also selected to improve recall. The research challenges identified in the phrase mapping step are: filling the lexical/vocabulary gap and multilinguality. The latter applies if the vocabulary in the user query and the KB vocabulary are expressed (lexicalized) in different languages. See section 13 for a transversal view on the research challenges.

8. Disambiguation

Two ambiguity problems can arise. The first is that from the question analysis step the segmentation and the dependencies between the segments are ambiguous. For example, in the question "Give me all european countries." the segmentation can group or not the expression "european countries" leading to two possibilities. The second is that the phrase mapping step returns multiple possible resources for one phrase. In the example

[14] https://github.com/dbpedia/lookup
[15] http://wikipedia-miner.cms.waikato.ac.nz

	Knowledge base labels	String similarity	Lucene index or similar	WordNet/Wiktionary	Redirects	PATTY	Using extracted knowledge	BOA or similar	Distributional Semantics	Wikipedia specific approaches	Reference
BELA	x	x	x	x	x				x		(Walter et al. 2012)
CASIA	x				x		x		x		(He et al. 2014)
DEANNA	x										(Yahya et al. 2012)
FREyA	x	x		x							(Damljanovic et al. 2010)
gAnswer	x				x		x			x	(Zou et al. 2014)
GFMed	x				x						(Marginean 2017)
Hakimov et al.	x							x			(Hakimov et al. 2015)
Intui2	x										(Dima 2013)
Intui3	x			x	x					x	(Dima 2014)
ISOFT	x		x			x			x		(Park et al. 2014)
POMELO	x										(Hamon et al. 2014)
PowerAqua	x		x	x	x						(Lopez et al. 2012)
QAKiS	x							x			(Cabrio et al. n.d.)
QAnswer	x	x	x	x	x			x			(Ruseti et al. 2015)
RTV	x		x						x		(Giannone et al. 2013)
SemGraphQA	x		x	x	x						(Beaumont et al. 2015)
SemSeK	x		x	x	x				x		(Aggarwal & Buitelaar 2012)
SINA	x										(Shekarpour et al. 2015)
SWIP	x	x									(Pradel et al. 2012)
TBSL	x	x	x	x				x			(Unger et al. 2012)
TR Discover	x										(Song et al. 2015)
Treo	x		x						x		(Freitas & Curry 2014)
UTQA	x	x			x				x		(Pouran-ebn veyseh 2016)
Xser	x					x				x	(Xu et al. 2014)
Zhang et al.	x	x									(Zhange et al. 2016)
Zhu et al.	x								x	x	(Zhu et al. 2015)

Table 4. Each line of this table corresponds to a QA system. The columns indicate which of the techniques presented in the phrase mapping task is used. The columns refer to the subsections of section 7.

above "european" could map to different meanings of the word "Europe". This section explains how question answering systems deal with these ambiguities.

8.1. Local Disambiguation

Due to ambiguities, QA systems generate many possible interpretations. To rank them mainly two features are used. The first is the (string or semantic) similarity of the label of the resource and the corresponding phrase. The second is a type consistency check between the properties and their arguments. The first feature is used to rank the possible interpretations, the second to exclude some. These features are "local" in a sense that only the consistency between the two resources that are directly related is checked. The advantage is that "local" disambiguation is very easy and fast. Moreover, it is often very powerful. A main disadvantage is that actual KBs often do not contain domain and range information of a property so that the type consistency cannot be done. Consider for example the question "Who is the director of The Lord of the Rings?". In this case "The Lord of the Rings" is clearly referring to the film and not to the book. If the property corresponding to director has no domain/range information then this strategy would not allow to decide if "The Lord Of the Rings" is a book or a film. The interpretation as a

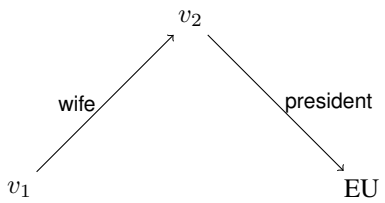

Figure 8. A graph G which is semantically equivalent to the question "Who is the wife of the president of the EU?". The ambiguity of "EU" in the phrase carries over to an ambiguity of the corresponding vertex of G.

book can only be excluded later by querying the KB. This form of disambiguation is used by all systems.

8.2. Graph search

The QA systems gAnswer (Zou et al. 2014), PowerAqua (Lopez et al. 2012), SemSek (Aggarwal & Buitelaar 2012) and Treo (Freitas & Curry 2014) use the graph structure of the KB to resolve the ambiguity although they follow two different strategies. We explain them using the following question as an example: "Who is the wife of the president of the EU?".

The QA system gAnswer (Zou et al. 2014) assumes that in the question analysis step the intention of a question q can be translated into a graph G. See Figure 8 for the question above. This contains a node for each variable, entity and class in the question, and an edge for each relation. Moreover, it is semantically equivalent to q. The ambiguity that arises in the phrase mapping step carries over to an ambiguity of vertices and edges of G. The idea of gAnswer is to disambiguate the vertices and edges of G by searching into the KB a subgraph isomorphic to G such that the corresponding vertices correspond to the segments of q with high probability. This is achieved by assigning a score to each possible match, which is proportional to the distance between the labels of the resources and the segments of the question. The top-k matches are retrieved. In a similar fashion, PowerAqua explores the candidate properties, however it uses an iterative approach to balance precision and recall, selecting first the most likely mappings and interpretations based on the question analysis, and re-iterating until an answer is found or all the solution space has been analysed. It assigns a score to each of the queries based on the score of the selected matches and if the matches are directly related or not (semantic distance).

The QA systems SemSek and Treo solve the ambiguity from another perspective. In this case, only the instances identified in the question analysis step are expanded in the phrase mapping step leading to ambiguity. The relational phrases are not expanded. For the concrete example "EU" would be identified and some candidate instances would be generated. The relational phrases "president of" and "wife of" are not expanded. Instead a graph search is started from the instances and all the properties attached to them are compared to the relational phrases detected in the question. In this case, all properties attached to the different interpretations of "EU" are compared. If no relation fits then the expanded instance is excluded.

Note that while the second approach has higher recall since all attached properties are explored, though, the first has probably higher precision.

The approach used in SemSek, Treo and gAnswer assume that one can deduce all relational phrases from the question. However, they can also be implicit. PowerAqua, is

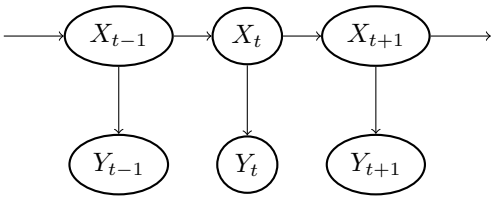

Figure 9. Conditional dependencies in a Hidden Markow Model

able to find a subgraph to translate the user query, even if not all entities in the query are matched. For all of them, the performance will decrease if there is too much ambiguity, i.e. if there are too many candidate matches to analyse.

8.3. Hidden Markov Model (HMM)

Hidden Markov Models (HMM) are used by SINA (Shekarpour et al. 2015) and RTV (Giannone et al. 2013) to address the ambiguity that arises in the phrase mapping phase. The idea behind this strategy is sketched using an example. Assume the question is: "By which countries was the EU founded?". In a Hidden Markov Model (HMM) one has two stochastic processes $(X_t)_{t\in\mathbb{N}}$ and $(Y_t)_{t\in\mathbb{N}}$ where only the last one is observed. The possible values of the random variables X_t are called hidden states whereas the possible values of the random variables Y_t are called observed states. For the example above, the set of observed states is *{"countries", "EU", "founded"}*, i.e. the set of segments of the question that have an associated resource. The set of hidden states is *{dbo:Country, dbr:Euro, dbr:European_Union, dbr:Europium, dbp: founded, dbp:establishedEvent}*, i.e. the set of possible resources associated with the segments.
In a Hidden Markov Model the following dependency between the random variables are assumed:

- $P(X_t = x_t|X_0 = x_0; ...; X_{t-1} = x_{t-1}; Y_0 = y_0; ...; Y_t = y_t) = P(X_t = x_t|X_{t-1} = x_{t-1})$, i.e. the value of a variable X_t only depends from the value of X_{t-1} which means that $(X_t)_{t\in\mathbb{N}}$ is a Markov Chain;
- $P(Y_t = y_t|X_0 = x_0; ...; X_t = x_t; Y_0 = y_0; ...; Y_{t-1} = y_{t-1}) = P(Y_t = y_t|X_t = x_t)$, i.e. that the value of Y_t depends only from X_t.

If one indicates the conditional dependencies with an arrow one gets the diagram in figure 9.

In the context of disambiguation this means that the appearance of a resource at time t depends only on the appearance of another resource at time $t-1$ and that the segments appear with some probability given a resource. The disambiguation process is reduced to the case of finding the most likely sequence of hidden states (the resources) given the sequence of observed states (the segments). This is a standard problem that can be solved by the Viterbi Algorithm.
To complete the modeling one needs to indicate three more parameters:

- the initial probability, i.e. $P(X_0 = x)$ for $x \in X$;
- the transition probability, i.e. $P(X_t = x_1|X_{t-1} = x_2)$ for $x_1, x_2 \in X$;
- the emission probability, i.e. $P(Y_t = y|X_t = x)$ for $x \in X$ and $y \in Y$.

These can be bootstrapped differently. In SINA the emission probability is set accord-

ing to a string similarity measure between the label of the resource and the segment. In RTV the emission probabilities are estimated using word embeddings. In SINA the initial probabilities and the transition probabilities are estimated based on the distance of the resources in the KB and their popularity. Retrieving the distance between all resources is computationally expensive making this approach slow. In RTV the initial and transition probabilities are set to be uniform for all resources making the estimation fast but more inaccurate.
An advantage of this technique is that it is not necessary to know the dependency between the different resources but only a set of possible resources.

8.4. Integer Linear Program (ILP)

In the QA system DEANNA (Yahya et al. 2012) it was proposed to set up an Integer Linear Program (ILP), which is an optimisation tool, to solve the disambiguation task. This technique addresses the ambiguity of the phrase mapping phase and some ambiguity that arises during the segmentation.
The ILP uses decision variables to model the problem, of finding the best answer, as an optimisation problem. In particular, it uses boolean variables to indicate if a segment of the question is chosen or not, if a resource corresponding to a segment is chosen or whether a segment corresponds to a property or an instance. The constraints include conditions such that the chosen segments do not overlap, such that if a segment is chosen then one corresponding resource must be chosen and so on. The optimization function includes three terms. The first increases if the label of a resource is similar to the corresponding segment. The second increases if two selected resources often occur in the same context. The third tries to maximize the number of selected segments. Note that the second term makes the disambiguation process work. Thus, after solving the ILP the optimal values obtained point to the optimal answer, which is returned as the answer to the question.
The main disadvantage is that some dependencies between the segments have to be computed in the question analysis phase.

8.5. Markov Logic Network

The question answering system CASIA (He et al. 2014) uses a Markov Logic Network (MLN) for the disambiguation task. MLN is used to learn a model for choosing the right segmentation, for mapping phrases to resources, and grouping resources into a graph. The idea is to define some constraints using first-order logic formulas. MLN allows to consider some of them as hard constraints that must be fulfilled and others as soft constraints, i.e. if they are not satisfied a penalty is applied. In this case both the ambiguities that arise in the question analysis and phrase mapping stage are resolved.
Examples of hard constraints are: if a phrase of the question is chosen then one of the corresponding resources must be chosen, or that the chosen phrases cannot overlap. Examples for soft constraints are: if a phrase has a particular POS tag then it is mapped to a relation, the resource whose label is most similar to the corresponding phrase in the question must be chosen. The hard constraints of a MLN have a similar behavior as the constraints in a ILP while the soft constraints allow more flexibility. The penalty for the soft constraints are learned in a training phase.

The advantage of a MLN is that they allow more flexibility than an ILP in choosing the constraints. However, a training phase is needed.

8.6. Structured perceptron

The engine Xser (Xu et al. 2014) uses a structured perceptron to solve the disambiguation task. The idea is to consider features during disambiguation such as: the similarity of a phrase and the corresponding resource, the popularity of a label for a resource, the compatibility of the range and domain of a property with the types of the arguments and the number of phrases in the question that are in the same domain. In a training phase for each of the features f a weight w is computed such that the expected configuration z fulfills:

$$z = argmax_{y \in Y(x)} w \cdot f(x, y)$$

where x is the question and $Y(x)$ is the set of possible configurations of the resources and dependency relations. The configuration which maximizes the expression above is chosen. In this approach, the ambiguity that arises in the phrase mapping phase is resolved. However, a training phase is needed.

8.7. User feedback

There exists situations in which the QA engine cannot do the disambiguation automatically. This can happen because the disambiguation technique used by the engine does not suffice or because the question is really ambiguous. Therefore, some systems ask the user to resolve the ambiguity by choosing between some proposed resources. A system that relies heavily on user feedback for the disambiguation is Freya (Damljanovic et al. 2012).

8.8. Summary and research challenges for disambiguation

Table 5 gives an overview of the different techniques used by the QA systems for disambiguation. Note that the systems GFMed, POMELO, TRDiscover, Zhang et al. were evaluated on the QALD-4 task 2 and there the disambiguation problem does not really arise since the three inter-linked KBs are too small. Beyond the disambiguation techniques, the research challenges in this domain are the lexical ambiguity of the matches, the use of vague prepositions or verbs (have/be) instead of explicit relationships (for example: movies of Pedro Almodovar) that can convey different intepretations (in the example: directed, produced, starring, etc.). These challenges are listed in section 13 where they are discussed through the previous surveys on KB-based QA systems.

9. Query construction

In this section, we describe how each QA system construct a SPARQL query. A problem arises during the query construction, the so called "semantic gap". Assume for example that a user asks the question: "Which countries are in the European Union?". One would probably assume that in the KB there are triples like:

	Local disambiguation	Graph search	HMM	LIP	MLN	Structured perceptron	User feedback	Reference
BELA	x							(Walter et al. 2012)
CASIA	x				x			(He et al. 2014)
DEANNA	x			x				(Yahya et al. 2012)
FREyA	x						x	(Damljanovic et al. 2010)
gAnswer	x	x						(Zou et al. 2014)
GFMed	x							(Marginean 2017)
Hakimov et al.	x							(Hakimov et al. 2015)
Intui2	x							(Dima 2013)
Intui3	x							(Dima 2014)
ISOFT	x							(Park et al. 2014)
POMELO	x							(Hamon et al. 2014)
PowerAqua	x	x						(Lopez et al. 2012)
QAKiS	x							(Cabrio et al. n.d.)
QAnswer	x							(Ruseti et al. 2015)
RTV	x		x					(Giannone et al. 2013)
SemGraphQA	x							(Beaumont et al. 2015)
SemSeK	x	x						(Aggarwal & Buitelaar 2012)
SINA	x		x					(Shekarpour et al. 2015)
SWIP	x						x	(Pradel et al. 2012)
TBSL	x							(Unger et al. 2012)
Treo	x	x						(Freitas & Curry 2014)
TR Discover	x							(Song et al. 2015)
UTQA	?							(Pouran-ebn veyseh 2016)
Xser	x					x		(Xu et al. 2014)
Zhang et al.	x							(Zhange et al. 2016)
Zhu et al.	x							(Zhu et al. 2015)

Table 5. Each line of this table corresponds to a QA system. The columns indicate which of the techniques presented in the disambiguation task are used. The columns refer to the subsections of section 8. We put a question mark if it is not clear from the publication which technique was used.

```
dbr:Greece dbp:member dbr:European_Union .
dbr:France dbp:member dbr:European_Union .
```

But this is not the case, in DBpedia the requested information is encoded as:

```
dbr:Greece dct:subject dbc:Member_states_of_the_European_Union .
dbr:France dct:subject dbc:Member_states_of_the_European_Union .
```

So instead of a property "dbp:member" DBpedia uses the class "dbc:Member_states_of_ the_European _Union" to encode the information. The "semantic gap" refers to the problem that the KB encodes an information differently from what one could deduce from the question. This shows that in general it is impossible to deduce the form of the SPARQL query knowing only the question. Therefore, we classify the approaches for the query construction based on how the SPARQL query form is deduced. We distinguish between approaches where the SPARQL query form is based on templates, approaches where it is deduced from the question analysis phase, where it is deduced using machine learning techniques or where it is deduced using only semantic information. The last subsection describes the approach of SemSek and Treo that do not generate a SPARQL query.

9.1. Query construction using templates

Some engines use templates to generate the SPARQL query, i.e. a set of predefined queries with some slots that have to be filled. The system QAKiS (Cabrio et al. n.d.) restricts to select queries with only one triple. The system ISOFT (Park et al. 2014) uses a small set of templates to generate SPARQL queries: these include ASK queries over one triple, some simple SELECT queries with one or two triples and templates that use a COUNT, ORDER BY or FILTER expression containing only one triple. Also, PowerAqua (Lopez et al. 2012) assumes that the input question can be reduced to one or two linguistic triples (not more than two predicates) following a set of templates, then each linguistic triple is semantically matched into one or more KB triples that can be combined into a SELECT query. After some graph-based disambiguation the SPARQL query is constructed. The disadvantage is that not all questions can be treated using templates.

9.2. Query construction guided by information from the question analysis

Most of the systems start with the information obtained in the question analysis part and deduce from it the form of the SPARQL query.
Freya and Intui3 (Dima 2014) start from the segmentation of the question. In the phrase mapping phase some segments have an associated resource. These resources are then combined into triples respecting the order of the segments in the question. If necessary some additional variables between the identified segments are added, for example if one relation is followed by another relation.
DEANNA (Yahya et al. 2013) selects some triples phrase candidates in the question analysis phase using regular expressions over POS tags. These are mapped to resources in the phrase mapping phase. In the disambiguation phase the best phrase candidates and the best phrase mappings are chosen using a ILP. This returns a set of triples that is then used to construct a SELECT query.
The QA systems gAnswer, QAnswer, RTV and SemGraphQA start with a dependency tree. In gAnswer (Zou et al. 2014) first the relations and the associated arguments are deduced. A graph G is constructed which has an edge for each relation and a vertex for each argument. The graph G reflects the structure of the final SPARQL query. The relations and arguments are mapped to possible resources. The right resources are identified using the sub-isomorphism strategy described in section 8.2. Then the SPARQL query is constructed.
QAnswer (Ruseti et al. 2015) first scans the dependency tree to find subgraphs of tokens that correspond to some resources. This way many graphs with associate resources are created. Then a local disambiguation is performed. The top ranked graph is chosen and the SPARQL query is constructed from this graph.
RTV (Giannone et al. 2013) uses the dependency graph to construct an ordered list of alternating properties and non properties. The corresponding resources are searched and disambiguated using a HMM. From this sequence the SPARQL query is generated.
Xser (Xu et al. 2014) uses three different machine learning algorithms. The first and the second are claimed to be KB independent. The first is used to determine the segments of the question corresponding to variables, properties, instances, and classes. The second is used to find the dependencies between the phrases as described in section 6.3.2. The third is used in the disambiguation phase, which is described in section 8.6 and is KB dependent. Since the first two algorithms are claimed to be KB independent this approach also constructs the form of the SPARQL by analyzing the question.

[H]

Lexical item	Syntactic category	Semantic representation
Barack Obama	NP	dbr : Barack_Obama
is	$(S\backslash NP)/(S\backslash NP)$	λf.λx.f(x)
married to	$(S\backslash NP)/NP$	λy.λx.dbo : spouse(x, y)
Michelle Obama	NP	dbr : Michelle_Obama

All these approaches share the same disadvantage. All of them make the implicit assumption that it is possible to deduce the structure of the SPARQL query from the structure of the question without knowing how the knowledge is encoded into the KB.

9.3. Query construction using Semantic Parsing

Semantic parsers are a particular type of parsers that couple syntactic rules to semantic composition. After parsing the question one gets a semantic interpretation of it.
From among the QA systems evaluated over QALD, different grammars for semantic parsers were used: GF grammars used by GFMed (Marginean 2017), feature-based context-free grammar (FCFG) used by TR Discover (Song et al. 2015), Combinatory Categorial Grammar (CCG) used by Hakimov et al. (Hakimov et al. 2015) and lexical tree-adjoint grammars (LTAG) used by TBSL (Unger et al. 2012) and BELA (Walter et al. 2012). We give a brief example using the CCG grammar. Consider the phrase "Barack Obama is married to Michelle Obama". To parse the sentence the following grammar rules are needed:

The first column indicates the phrases to which the rules are associated. The main syntactic categories are NP standing for noun phrase and S standing for sentence and combinations of them. The syntactic category $(S\backslash NP)/NP$ for example indicates that it can be combined with a noun phrase (NP) on the left and on the right to get a sentence S. Applying these rules, the sentence can be parsed from a syntactic point of view. Coupled to the syntactic rules is a semantic representation. Without going into details this is expressed using lambda calculus. For example, the phrase *married to* semantically is a binary function which takes two arguments. Since it is the passive form of the property `dbo:spouse` the arguments are inverted. The semantic representation of *Michelle Obama* is just a constant which in DBpedia is `dbr:Michelle Obama`. The application of the above syntactic rule between *married to* and *Michelle Obama* results in the semantic representation $\lambda x.dbo : spouse(x, dbr : Michelle\_Obama)$, i.e. *x* is the spouse of *Michelle Obama*. This way the whole sentence can be parsed leading to the semantic representation $dbo : spouse(dbr : Barack\_Obama, dbr : Michelle\_Obama)$, i.e. *Barack Obama*'s spouse is *Michelle Obama*. This way the sentence is completely understood from a semantic point of view as well. The semantic representation can be translated then to a SPARQL query.
The main advantage of this approach is that one can directly get a semantic representation of a question. This also includes the superlative and comparative variations of the question. A disadvantage is that the questions have to be well-formulated, i.e. they are not robust with respect to malformed questions. The main disadvantage is that one needs to have for each lexical item a corresponding semantic representation. To generate the semantic representations, Hakimov et al. (Hakimov et al. 2015) adapted the algorithm of Zettlemoyer & Collins (Zettlemoyer & Collins 2012) that generates the semantic representation from a learning corpus of pairs of questions and the corresponding semantic representation. The main problem encountered by Hakimov et al. (Hakimov et al. 2015) is that many lexical items do not appear in the training corpus leading to low recall. To

alleviate this problem, TBSL (Unger et al. 2012) generates candidate semantic representations of unknown lexical items based on their POS tags. In the example above, if the lexical item *married to* is unknown then possible interpretations are generated such as two binary functions with the arguments x and y exchanged. Since there is no knowledge about the fact that *married to* has to be mapped to the property $dbo : spouse$, thus some templates are generated, i.e. the parser is able to parse the question, but a binary property corresponding to *married to* has still to be found.

9.4. Query construction using machine learning

CASIA (He et al. 2014) uses a machine learning approach for the whole QA process. The question analysis phase is used to segment the question and to extract features like the position of a phrase, the POS tag of a phrase, the type of dependency in the dependency tree and some other. In the phrase mapping phase resources are associated with the segments and new features are extracted: the type of the resources and a score for the similarity between the segment and the resource. In the disambiguation phase the extracted features are used in a MLN (as described in 8.5) to find the most probable relation between the segments and to find the most probable mapping. The detected relations are then used to generate the SPARQL query. The disambiguation phase must be retrained for each new KB.

9.5. Query construction relaying on semantic information

The QA system SINA (Shekarpour et al. 2015) was developed to deal primarily with keyword queries, i.e. instead of inserting the question "What is the capital of Belgium?" the user can also insert the keywords "capital Belgium". This implies that the relation between the different resources is not explicit like in a natural language question. In this case the dependencies between the resources have to be derived using the KB.
SINA first segments the question and finds associated resources. These are disambiguated using a HMM. Once the resources are determined, SINA constructs the query in the following way. For each instance or class, a vertex is created. For each property, an edge is created. The edges are used to connect the vertices if the types of the range and domain allow it. If not, then either one or two new vertices are created that correspond to variables. Note that the combinatorics could allow more than one graph. In this case, they are all considered because it is not clear which one reflects the user's intention. Moreover, at the end of the process, it is possible that one gets unconnected subgraphs. In this case, for each pair of vertices in two fixed subgraphs the set of possible properties that can connect them is computed and taken into account. All the possible graphs are translated to a SPARQL query and executed. The QA system POMELO proceeds in a similar way. The QA system developed by Zhang et al. also does not rely on syntactic features to construct the query. The query is generated using an ILP. First, those resources are identified which can be referred to by the question. These are then combined using some hand made rules into triple patterns. Between all possible triple patterns some are selected using an ILP.
The advantage of this strategy is that the graph is constructed starting from the underlying KB and not using the syntax of the question. The disadvantages are that this process is computationally complex and that the syntax of the question is not respected. For example, the systems will not see any difference between the questions "Who is the mother of Angela Merkel?" and "Angela Merkel is the mother of who?".

9.6. Approach not using SPARQL

Treo (Freitas & Curry 2014) and SemSek (Aggarwal & Buitelaar 2012) do not generate a SPARQL query, but instead navigate through the KB by dereferencing resources. Consider the following example: "In which city was the leader of the European Union born?". SemSek first identifies a "central term" in the question, in this case "European Union". Using the dependency tree and starting from the "central term" an ordered list of potential terms corresponding to resources is generated. In the example above the list would be: "European Union", "leader", "born", "city". Then candidate resources for the first term (for example dbr:European_Union) are searched and the corresponding URI http://dbpedia.org/resource /European_Union is dereferenced to search all corresponding properties. These are compared with the second term in the list. The object is considered if one of the property is similar (as a string or semantically) to the second term in the list (like dbp:leaderName or dbp:leaderTitle). This generates two new exploring directions in the algorithm. Otherwise the actual direction is not further analyzed. The systems continue like this and search the right answer in the graph.

9.7. Summary and research challenges for the query construction task

Table 6 gives an overview of the different techniques used by the QA systems for the query constructing. In this domain, the current research challenges that are identified are the interpretation of adjective modifiers and superlatives; the implementation of aggregation, comparison and negation operators; the resolution of syntactic and scope ambiguities; the non-compositionality, and the semantic tractability. Note that all challenges identified in previous surveys are discussed in section 13.

10. Querying distributed knowledge

In the previous sections, we discussed the techniques for answering questions over a single KB. Nevertheless, one pertinent question would be: what changes in the case of multiple KBs? Only few QA systems tackled this problem. We found in the literature that the problem can be classified into two groups. The first assumes that the KBs are disjoint graphs. The second assumes that the KBs are interlinked, i.e. resources that refer to the same entity are identified through the KBs using owl:sameAs links (two identified resources are called aligned).

10.1. Considering unconnected KBs

In this setting the KBs are not interlinked and in particular different KBs can refer to the same entity using different URIs. This scenario was tackeled by PowerAqua (Lopez et al. 2012) and Zhang et al (Zhange et al. 2016). Assume for example that for the question "Which rivers flow in European countries?" a part of the information can be found into two different KBs. One containing information like "(?river, flow, ?country)" and the second one "(?country, type, European)" (i.e in general one can say that there are triple-patterns matching different KBs). These cannot be executed as a SPARQL query because the URIs for countries in the first and second KBs are different and not linked. Therefore, the results are retrieved independently from both KBs and merged by comparing the labels of the URIs. Then either a SPARQL query is generated that

	Using templates	Using info. from the QA	Using Semantic Parsing	Using machine learning	Semantic information	Not generating SPARQL	Reference
BELA			x				(Walter et al. 2012)
CASIA				x			(He et al. 2014)
DEANNA		x					(Yahya et al. 2012)
FREyA		x					(Damljanovic et al. 2010)
gAnswer		x					(Zou et al. 2014)
GFMed			x				(Marginean 2017)
Hakimov et al.			x				(Hakimov et al. 2015)
Intui2		x					(Dima 2013)
Intui3		x					(Dima 2014)
ISOFT	x						(Park et al. 2014)
POMELO					x		(Hamon et al. 2014)
PowerAqua	x						(Lopez et al. 2012)
QAKiS	x						(Cabrio et al. n.d.)
QAnswer		x					(Ruseti et al. 2015)
RTV		x					(Giannone et al. 2013)
SemGraphQA		x					(Beaumont et al. 2015)
SemSeK						x	(Aggarwal & Buitelaar 2012)
SINA					x		(Shekarpour et al. 2015)
SWIP	x						(Pradel et al. 2012)
TBSL			x				(Unger et al. 2012)
Treo						x	(Freitas & Curry 2014)
TR Discover			x				(Song et al. 2015)
UTQA	?						(Pouran-ebn veyseh 2016)
Xser		x					(Xu et al. 2014)
Zhang et al.					x		(Zhange et al. 2016)
Zhu et al.						x	(Zhu et al. 2015)

Table 6. Each line of this table corresponds to a QA system. The columns indicate which of the techniques presented in the query construction task are used. The columns refer to the subsections of section 9.

contains the aligned URIs (i.e. uri:a owl:sameAs uri:b) or the result is computed without a SPARQL query.

10.2. Considering interlinked KBs

The task of querying interlinked KBs was proposed at QALD-4. It was tackled by GFMed (Marginean 2017), SINA (Shekarpour et al. 2015), POMELO (Hamon et al. 2014), TR Discover (Song et al. 2015) and Zhang et al (Zhange et al. 2016). In the case where the KBs are interlinked there is no particular technique used. In fact one can see the interlinked KBs as one big KB. The identification with `owl:sameAs` links must be considered during query construction. Note that, in such a scenario, scalability can easily become a problem. This is not the case for the QALD-4 task since only three small KBs are interlinked.

	Unconnected KBs	Interlinked KBs	Reference
GFMed		x	(Marginean 2017)
POMELO		x	(Hamon et al. 2014)
PowerAqua	x		(Lopez et al. 2012)
SINA		x	(Shekarpour et al. 2015)
TR Discover		x	(Song et al. 2015)
Zhang et al.	x		(Zhange et al. 2016)

Figure 10. Each line of this table corresponds to a QA system and indicates which of the techniques presented in the distributed task it uses. The columns refer to the subsections of section 10.

QA systems	Precision	Recall	F-measure	Reference
Yao et al. (2014)	0.480	0.337	0.354	(Yao & Van Durme 2014)
SEMPRE	0.413	0.480	0.357	(Berant et al. 2013)
Bao et al. (2014)	-	-	0.375	(Bao et al. 2014)
Bordes et al. (2014)	-	-	0.392	(Bordes et al. 2014)
PARASEMPRE	0.466	0.405	0.399	(Berant & Liang 2014)
Clarke et al. (2015)	-	-	0.401	(Clarke 2015)
Dong et al. (2015)	-	-	0.401	(Dong et al. 2015)
Yang et al. (2014)	-	-	0.413	(Yang et al. 2014)
GraphPharser	-	-	0.413	(Reddy et al. 2014)
Bordes et al. (2015)	-	-	0.422	(Bordes et al. 2015)
Zhang et al. (2016)	-	-	0.426	(Zhang et al. 2016)
Yao (2015)	0.545	0.526	0.443	(Yao 2015)
Yang et al. (2015)	-	-	0.449	(Yang et al. 2015)
Aqqu	0.604	0.498	0.494	(Bast & Haussmann 2015)
AgendaIL	-	-	0.497	(Berant & Liang 2015)
Wang et al. (2014)	0.525	0.447	0.453	(Wang et al. 2014)
Reddy et al. (2016)	0.611	0.490	0.503	(Reddy et al. 2016)
Abujabal et al. (2017)	-	-	510	(Abujabal et al. 2017)
Yavuz et al. (2016)	-	-	0.516	(Yavuz et al. 2016)
STAGG	0.607	0.528	0.525	(Yih et al. 2015)
Ture et al. (2016)	-	-	0.522	(Ture & Jojic 2016)
Jain (2016)	0.649	0.552	0.557	(Jain 2016)

Table 7. This table summarizes the QA systems evaluated over WebQuestions. It contains publications describing a QA system evaluated over WebQuestions that cited (Berant et al. 2013) according to google scholar.

11. QA systems evaluated over WebQuestions

In this section, we describe QA systems evaluated over WebQuestions. Table 7 contains a list of QA systems evaluated over WebQuestions. To identify these systems, we searched in Google Scholar for all publications citing the publication (Berant et al. 2013) which introduced the benchmark. In the following, we focus on techniques used by QA systems evaluated over WebQuestions that were not used by QA systems evaluated over QALD. Due to space limitations, a detailed analysis as for QALD is not possible.

Many of the techniques presented in section 6 about question analysis are also used by QA systems evaluated over WebQuestions. For example, Reddy et al. 2016 makes an ex-

tensive use of dependecy trees, while SEMPRE (Berant et al. 2013) and PARASEMPRE (Bordes et al. 2014) use POS tags as features. In addition to the techniques presented in section 6 some works make use of neural networks. These have been successfully used in many NLP areas. Different architectures have been explored. Convolutional neural networks and recurrent neural networks where used to identify or focus on particular parts of the questions (like the one referring to the type, the relation or the entity contained in a question). Convolutional neural networks are for example used in Dong et al. 2015 while recurrent neural networks are used in Zhang et al. 2016 and Ture & Jojic 2016.

Many QA system use similar approaches to the one presented in section 7 about the pharse mapping task. For example the strategy presented in section 7.3.5 is used by SEMPRE (Berant et al. 2013), Yang et al. 2015 and GraphPharser (Reddy et al. 2014).

One additional dataset that was used to close the lexical gap was the PARALEX corpus (Fader et al. 2013). It contains 18 million pairs of questions from wikianswers.com which where tagged as having the same meaning by users. This dataset can be used to learn different ways to express the same relation. It was used for example by PARASEMPRE (Bordes et al. 2014) and Bordes et al. 2015. Moreover differently from QALD many works use the datasets and strategies presented in section 7 to create embeddings (using neural networks) for relations and entities. This is for example the case for Bordes et al. 2015, Yang et al. 2014, Yavuz et al. 2016, STAGG (Yih et al. 2015) and Ture & Jojic 2016.

The disambiguation problem discussed in section 8 is approached with similar strategies both in QALD and WebQuestions. To disambiguate the entities, many QA systems evaluated over WebQuestions use EL tools for Freebase. To disambiguate between different generated queries, mainly machine learning algorithms are used like structured perceptrons, Reddy et al. 2016, and learning to rank used in Aqqu Bast & Haussmann 2015. The features are mainly the same.

The query construction problem discussed in section 9 is simpler than for QALD. Many QA systems restrict to simple triple patterns like Bordes et al. 2015, Ture & Jojic 2016 while most of them restrict to simple or reified triple patterns like it is done in Aqqu (Bast & Haussmann 2015) or Reddy et al. 2016. Some QA systems follow a strategy that is similar to the one presented in section 9.6. However, one strategy is not used over QALD, it is generally referred to "information retrieval approach". Consider for example the question: "When did Avatar release in UK?". First, the main entity of the question is searched. In the example, the entity is "Avatar". Then all entities connected to it via a simple or a reified statement are considered as a possible answer. This includes for example the date "17-12-2009" but also the director of Avatar, the characters that played in the film and many others. In this manner, the problem is reduced to a classification problem, i.e., determining if an entity is an answer or not. This is done, for example, by Yao & Van Durme 2014, Bordes et al. 2014, Bordes et al. 2015, Dong et al. 2015 and Zhang et al. 2016.

The task of querying distributed knowledge is not addressed in the WebQuestions benchmark.

In summary, one can say that over WebQuestions more attention was put on closing the lexical gap while in QALD more effort was done in the query construction process. This is mainly due to the type of questions contained in the benchmarks.

QA systems	F-measure	Reference
Bordes et al. (2015)	0.627	(Bordes et al. 2015)
Yin et al. (2016)	0.683	(Yin et al. 2016)
Dai et al. (2016)*	0.626	(Dai et al. 2016)
Golub and He (2016)	0.709	(Golub & He 2016)
Lukovnikov et al. (2017)	0.712	(Lukovnikov et al. 2017)

Table 8. This table summarizes the QA systems evaluated over SimpleQuestions. It contains publications describing a QA system evaluated over SimpleQuestions that cited (Bordes et al. 2015) according to google scholar. Every system was evaluated over FB2M except the one marked with $(*)$ which was evaluated over FB5M.

12. QA systems evaluated over SimpleQuestions

In this section, we describe QA systems evaluated over SimpleQuestions. Table 8 contains a list of QA systems evaluated over SimpleQuestions. To identify these systems, we searched in Google Scholar for all publications citing the publication (Bordes et al. 2015) which introduced the benchmark.
All systems evaluated over this dataset use neural networks architectures. In fact this benchmark was created to experiment with these machine learning technique since it needs a lot of training data to perform well. The different QA Systems evaluated over SimpleQuestions follow a similar strategy. Remember that every question in this benchmark can be solved by one single triple, i.e. only a subject and a predicate has to be found. The possible triples are found in the following way. First using an n-gram strategy (see section 6.1) candidate entities are identified. Then by looking in the KB all triples containing these entities as a subject are identified. The problem is reduced to choose between the list of triples the right one. This last step is done using different neural network architectures. Let's see more in detail how the different phases of the QA process are tackled.
The question analysis task is generally not treated as a separate task but some parts of the networks are dedicated to it. Golub & He 2016 and Yin et al. 2016 use the so called attention mechanisms. This means that the network learns to focus on the parts of the questions that refer to a subject or to a predicate.
For the phrase mapping task the QA systems relay on the labels provided by the KB as described in subsection 7.1. Moreover the question is encoded either at the character level like in Golub & He 2016 or, using word embeddings like in Dai et al. 2016 or, in both ways like in Lukovnikov et al. 2017. Word embedding is used to bridge the lexical gap following the idea presented in subsection 7.3.5. The character encoding is used to overcome the out-of-vocabulary problem, i.e. the number of possible words appearing in a question is too big to include them all in the vocabulary.
The disambiguation problem is handled as a ranking problem. Here the different neural network architectures are used to compute a score between a pair of a question, and a tuple of a subject and a predicate. The pair with the highest score is token.
The query construction problem is very simple since only SPARQL queries with one triple pattern must be generated. The problem of querying multiple KBs is not addressed.
The main problem tackled in this benchmark is the disambiguation problem.

Challenge	Survey, Years covered	Short description	Task
Identify Question types	(Cimiano & Minock 2009), 2004-07, (Lopez et al. 2013), 2005-12	Includes wh-questions, requests (*give me*), nominal or definitions, topicalised entities, how questions with adjective or quantifications (*how big/many*)	Question Analysis
Lexical gap, Vocabulary gap	(Lopez et al. 2011), 2004-11, (Lopez et al. 2013), 2005-12, (Freitas et al. 2012), 2004-11, (Freitas et al. 2015), 2011-12, (Höffner et al. 2016), 2011-15	Query and databases may not be expressed using the same vocabulary (synonymy) or at the same level of abstraction. It requires to bridge the gap between the vocabulary in the user query and the KB vocabulary	Mapping
Multilingual QA	(Cimiano et al. 2013), QALD-3, (Unger et al. 2014), QALD-4, (Höffner et al. 2016), 2011-15	Mediates between a user expressing an information need in her own language and the semantic data	Question Analysis, Mapping
Light expression Vagueness Lexical ambiguity	(Cimiano & Minock 2009), 2004-07, (Lopez et al. 2011), 2004-11, (Lopez et al. 2013), 2005-12, (Freitas et al. 2015), 2004-11, (Höffner et al. 2016), 2011-15	Queries with words that can be interpreted through different ontological entities or semantically weak constructions. Relations that are expressed implicitly with the use of verbs such as *be/have* or light prepositions that can convey different meanings	Disambiguation
Semantic gap Conceptual complexity	(Lopez et al. 2013), 2005-12, (Freitas et al. 2012), 2004-11, (Freitas et al. 2015), 2011-12	Queries that are not necessarily structured in the knowledge base in the same way than in the question.	Query construction
Spatial and temporal prepositions	(Cimiano & Minock 2009), 2004-07, (Lopez et al. 2013), 2005-12, (Höffner et al. 2016), 2011-15	Requires to capture the domain-independent meaning of spatial (*in, next, thorough*) and temporal (*after, during*) prepositions	-
Adjective modifiers and superlatives	(Cimiano & Minock 2009), 2004-07, (Lopez et al. 2011), 2004-11, (Lopez et al. 2013), 2005-12, (Höffner et al. 2016), 2011-15	Superlative modifiers and attribute selectors (*how+adj*) require mapping each adjective to a KB predicate (e.g.,*area/ population* for *smallest*), as well as keeping the polarity for superlatives (order by ASC/DESC)	Disambiguation, Query construction
Aggregations, comparison and negation operators	(Cimiano & Minock 2009), 2004-07, (Lopez et al. 2011), 2004-11, (Lopez et al. 2013), 2005-12, (Höffner et al. 2016), 2011-15	Aggregation operators are those calculating a min, max, sum, an average or a count over a number of individuals fulfilling a certain property. Comparison operators compare numbers to a given order. The challenge is to realize a quantifier through logical operators	Question Analysis, Query Construction
Syntactic and Scope ambiguities	(Cimiano & Minock 2009), 2004-07	Syntactic ambiguity regarding the constituent that prepositional phrases or relative clauses can attach to (to the last or to a non-precedent constituent in a sentence), or when multiple scope quantifiers are present in a query (*most, all, each, etc.*)	Query construction
Distributed QA Entity reconciliation	(Lopez et al. 2011), 2004-11, (Freitas et al. 2012), 2004-11, (Unger et al. 2014), QALD-4, (Höffner et al. 2016), 2011-15	Combining facts across sources requires matching at schema level as well as entity level (find semantically equivalent dataset entities given a query entity) to join partial results or translations	Distributed Knowledge
Non-compositionality	(Cimiano & Minock 2009), 2004-07	Parts of a question that do not correspond to any logical form and need to be ignored (e.g., *largest cities in the world* if *world* is not explicitly model)	Query construction
Semantic tractability	(Freitas et al. 2012), 2004-11	To answer queries not supported by explicit dataset statements (e.g., inferring an entity *x* is an *actress* because of the statement *x starred y movie*)	Mapping, Disambiguation, Query construction
Out of scope	(Cimiano & Minock 2009), 2004-07, (Lopez et al. 2011), 2004-11	A system should inform about the fact that the question is out of scope of the KB (vs. out of the capabilities of the system)	Across all
Portability	(Cimiano & Minock 2009), 2004-07, (Lopez et al. 2011), 2004-11	The level of effort require to port the system to other sources (e.g., handcrafted lexicon, training)	Across all
Scalability	(Lopez et al. 2011), 2004 - 2011	Required both in terms of KB size and their number while keeping real time performance	Across all
Hybrid QA, Semantic and textual gap	(Lopez et al. 2011), 2004-11, (Unger et al. 2014), QALD-4	Combining both structured and unstructured information into one answer	-

Table 9. Challenges in the state of the art for QA systems over KBs

13. Evolution of research challenges of Question Answering over KBs

We look at the research challenges described in the literature since the beginning of open domain KB-based QA, through different published surveys and QALD evaluation reports. Table 9 lists each challenge, together with the reference(s) to the survey where it

is discussed and the years covered, i.e., the range of publication years for the KB-based QA systems cited in the given survey, excluding other kind of QA systems that share some of the challenges (such as Natural Language Interfaces for Databases). We group similar challenges together and, if it applies, map them across the five tasks in the QA process. As such we associate the tasks, in which we cluster the techniques analysed in this paper, with the challenges they try to tackle.

The first survey by Cimiano and Minock (Cimiano & Minock 2009) published in 2010 presents a quantitative analysis based on the input (user) questions as well as the outputs (formal queries) from a Geographical dataset, often used to evaluate the first domain-specific KB-based systems. The following challenges are identified: question types; language light; lexical ambiguities; syntactic ambiguities; scope ambiguities; spatial and temporal prepositions; adjective modifiers and superlatives, aggregation, comparison and negations; non-compositionality; out of scope; and variability of linguistic forms (i.e., questions matching to the same logical query - not included in Table 9).

Some of these first identified challenges, like capturing the meaning of spatial and temporal prepositions, are not yet tackled by the kind of QA systems surveyed here. Cimiano and Minock also argued that research on deep linguistic and semantic approaches, such as: paraphrase generation, discourse processing and guidance; could improve the performance of these systems by guiding users to generate a non-ambiguous form of the question (i.e., in line with the capabilities of the system), as well to process questions not in isolation, but in context with previously asked questions.

For these early systems, portability was one of the most challenging issues, although the customisation effort was rarely quantified or compared (Cimiano & Minock 2009).

Lopez et al. 2011 survey (Lopez et al. 2011) discusses the challenges that arised when moving from a classic KB system to an open domain QA system for the Web of data, in terms of portability, scalability, mapping and disambiguation, distributed query (fusion and ranking of facts across sources) and bridging the gap between the semantic data and unstructured textual information. Lexical ambiguities (noun homonymy or polysemy) are less of a problem when querying an unambiguous knowledge representation in a restricted domain. However, for open domain QA, a major challenge is to disambiguate the correct interpretation while scaling up to large and heterogeneous sources. Filtering and ranking techniques are often used to prune the large space of candidate solutions and obtain real time performance (Lopez et al. 2011).

Freitas et al. 2012 (Freitas et al. 2012) categorized the challenges on querying heterogeneous datasets into query expressivity, usability, vocabulary-level semantic matching, entity reconciliation and tractability. As stated in (Freitas et al. 2012) differently from IR, entity centric QA approaches aim towards more sophisticated semantic matching techniques to target queries with high expressivity (able to operate over the data, including superlatives, aggregators, etc.), without assuming that the users are aware of the internal representations (high usability through intuitive query interfaces). For Freitas et al. 2015 (Freitas et al. 2015) the process of mapping and semantic interpretation of schema agnostic queries involves coping with conceptual complexity, term ambiguity, vocabulary gap (synonymy) and vagueness / indeterminacy (where words or propositions fail to map the exact meaning intended by the transmitter). Freitas et al. (Freitas et al. 2015) proposed entropy measures to quantify the semantic complexity of mapping queries to database elements, showing that a large number of possible interpretations for words/prepositions had a negative impact in the F-measure reported for systems participating in QALD-1 and QALD-2.

Most of the challenges specified by Cimiano and Minock (Cimiano & Minock 2009) for the first domain specific QA systems remain valid for the open domain questions presented in the QALD benchmarks. Based on these challenges and the results from

QALD-1 and QALD-2 campaigns, Lopez et al. 2013 (Lopez et al. 2013) analyzed the characteristic problems involved in the task of mapping NL queries to formal queries, such as: (1) the limitations to bridge the lexical gap (between the vocabulary in the user query and the KB vocabulary) using only string metrics and generic dictionaries; (2) the difficulties arising on interpreting words through different entities because of the openness and heterogeneity of the sources (including duplicated or overlapping URIs and complex conjunctive KB terms from YAGO categories); and (3) complex queries with information needs that can only be expressed using aggregation (requiring counting), comparison (requiring filters), superlatives (requiring sorting results), temporal reasoning (e.g., on the same day, etc.), or a combination of them.
As well as the challenges reported in Lopez et al. 2013 (Lopez et al. 2013) for the first two campaigns, QALD-3 (Cimiano et al. 2013) introduced the multilinguality challenge and QALD-4 (Unger et al. 2014) introduced two new challenges as part of two new tasks. The first task, distributed QA, introduces questions for which answers are distributed across a collection of interconnected datasets in the biomedical domain. The second task, Hybrid QA, evaluates approaches that can process both structured and unstructured information, considering that lots of information is still available only in textual form, e.g., in the form of abstracts.
Lastly, the latest survey on challenges for semantic QA systems (Höffner et al. 2016) classifies these into seven challenges: the lexical gap, ambiguity, multilingualism, complex queries, distributed knowledge, temporal and spatial queries and templates. The templates challenge refer to the systems that use templates to capture the structure of (complex) queries with more than one basic graph pattern.

14. Conclusions

This analysis of QA systems shows that most of the systems have many common techniques. For example, a lot of systems use similar techniques in the question analysis and phrase mapping task. Moreover, it shows that, QA systems generally concentrate on improving only some techniques, whereas, they leave aside some others. For example, a lot of systems only use local disambiguation techniques, others only use basic techniques for the phrase mapping task.
It is nearly impossible to compare existing techniques individually. There are many reasons for that. From the performance of a monolithic system it is impossible to deduce the contribution of a single part. It is also not sufficient to just compare techniques that solve the same task against each other since a QA system is a combination of many components, and a balance between precision and recall must be found considering all steps. A component with high recall and low precision can be good or bad depending on the next steps. So a comparison can only be made by comparing the combinations of different techniques in a whole pipeline.
One solution to address these problems is to develop future QA systems using a modular approach. This will allow the community to contribute to new plug-ins in order to either replace existing approaches or create new ones. The aim is that a QA system should not always be build from scratch such that research can focus more on single tasks. As a side effect, it will become easier to compare techniques which solve the same task. These plug-ins should satisfy the constraints of the Semantic Web. They should be KB independent, scalable, possibly multilingual and require as few efforts as possible to set up a QA system over a new dataset. We are aware of four frameworks that attempt to provide a reusable architecture for QA systems. QALL-ME (Ferrandez

et al. 2011), openQA (Marx et al. 2014), OKBQA[16] and QANARY (Both et al. 2016, Diefenbach, Singh, Both, Cherix, Lange & Auer 2017). Such integration would also allow to build and reuse services around QA systems like speech recognition modules and reusable front-ends (Diefenbach, Amjad, Both, Singh & Maret 2017). This way the research community is enabled to tackle new research directions like developing speech recognition systems specifically designed to answer questions over KBs and study the interaction of QA systems with the user.
The comparison between QALD and WebQuestions shows that these benchmarks are quite similar. Despite that, both benchmarks are addressed by two quite isolated communities. That can be seen by the fact that QA systems are either evaluated on QALD or on WebQuestions and not on both. We hope that in future these communities will meet so that both can learn from each other. One possible meeting point is the Wikidata KB. There are several reasons for that. The Freebase KB is not updated anymore and all related services were shut down. The Wikidata dump contains both a reified and a non-reified version of the information so that moving from DBpedia and Freebase to Wikidata is simple. Moreover there are more and more well-maintained services around Wikidata that can be used to develop new QA systems.
At the same time, the fact that most QA systems are evaluated only over one KB, shows that more work is needed in creating QA systems that are truly KB independent. This would also allow to get new insights of how the quality of the KB affects the quality of the QA system. These point should be tackled more by the QA community.
Finally a poorly addressed point is the interaction of QA systems with users. We are not aware of works that study the usability of open domain KB-based QA systems and there are a few works applying QA over KBs in real scenarios (Song et al. 2015, Lopez et al. 2016). The interaction with the user can also be a good opportunity to improve QA systems over time and to collect training data which is becoming more and more important to improve the performance and understanding of these systems.
Research in QA over KBs is and remains a hot and interesting topic!

Acknowledgements. This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Sklodowska-Curie grant agreement No 642795.

References

Abujabal, A., Yahya, M., Riedewald, M. & Weikum, G. (2017), Automated template generation for question answering over knowledge graphs, *in* 'Proceedings of the 26th International Conference on World Wide Web', pp. 1191–1200.

Aggarwal, N. & Buitelaar, P. (2012), 'A system description of natural language query over dbpedia', *Proc. of Interacting with Linked Data (ILD 2012)[37]* .

Allam, A. M. & Haggag, M. H. (2012), 'The question answering systems: A survey', *Int Journal of Research and Reviews in Information Sciences (IJRRIS)* **2**(3), .

Atzori, M., Mazzeo, G. & Zaniolo, C. (2016), QA3@QALD-6: Statistical Question Answering over RDF cubes, *in* 'ESWC'. *to appear*.

Bao, J., Duan, N., Zhou, M. & Zhao, T. (2014), 'Knowledge-based question answering as machine translation', *Cell* **2**(6).

[16] http://www.okbqa.org

Bast, H. & Haussmann, E. (2015), More accurate question answering on freebase, *in* 'Proceedings of the 24th ACM International on Conference on Information and Knowledge Management', ACM.

Baudiš, P. & Šedivỳ, J. (2015), 'QALD challenge and the YodaQA system: Prototype notes'.

Beaumont, R., Grau, B. & Ligozat, A.-L. (2015), SemGraphQA@QALD-5: LIMSI participation at QALD-5@CLEF, *in* 'Working Notes for CLEF 2015 Conference', CLEF.

Berant, J., Chou, A., Frostig, R. & Liang, P. (2013), Semantic Parsing on Freebase from Question-Answer Pairs., *in* 'EMNLP'.

Berant, J. & Liang, P. (2014), Semantic parsing via paraphrasing., *in* 'ACL (1)'.

Berant, J. & Liang, P. (2015), 'Imitation learning of agenda-based semantic parsers', *Transactions of the Association for Computational Linguistics* .

Bordes, A., Chopra, S. & Weston, J. (2014), 'Question answering with subgraph embeddings', *arXiv preprint arXiv:1406.3676* .

Bordes, A., Usunier, N., Chopra, S. & Weston, J. (2015), 'Large-scale simple question answering with memory networks', *arXiv preprint arXiv:1506.02075* .

Both, A., Diefenbach, D., Singh, K., Shekarpour, S., Cherix, D. & Lange, C. (2016), Qanary–a methodology for vocabulary-driven open question answering systems, *in* 'International Semantic Web Conference', Springer.

Cabrio, E., Cojan, J., Aprosio, A. P., Magnini, B., Lavelli, A. & Gandon, F. (n.d.), QAKiS: an open domain QA system based on relational patterns, *in* 'Proceedings of the 2012th International Conference on Posters & Demonstrations Track-Volume 914', CEUR-WS. org.

Cimiano, P., Lopez, V., Unger, C., Cabrio, E., Ngomo, A.-C. N. & Walter, S. (2013), Multilingual question answering over linked data (qald-3): Lab overview, *in* 'Information Access Evaluation. Multilinguality, Multimodality, and Visualization', Springer.

Cimiano, P. & Minock, M. (2009), Natural language interfaces: What is the problem?-a data-driven quantitative analysis., *in* 'NLDB', Springer, pp. 192–206.

Clarke, D. (2015), 'Simple, fast semantic parsing with a tensor kernel', *arXiv preprint arXiv:1507.00639* .

Cunningham, H., Maynard, D., Bontcheva, K. & Tablan, V. (2002), GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications, *in* 'Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)'.

Dai, Z., Li, L. & Xu, W. (2016), 'Cfo: Conditional focused neural question answering with large-scale knowledge bases', *arXiv preprint arXiv:1606.01994* .

Daiber, J., Jakob, M., Hokamp, C. & Mendes, P. N. (2013), Improving efficiency and accuracy in multilingual entity extraction, *in* 'Proceedings of the 9th International Conference on Semantic Systems', ACM.

Damljanovic, D., Agatonovic, M. & Cunningham, H. (2010), Identification of the Question Focus: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction., *in* 'LREC'.

Damljanovic, D., Agatonovic, M. & Cunningham, H. (2012), FREyA: An interactive way of querying linked data using natural language, *in* 'The Semantic Web: ESWC 2011 Workshops', Springer.

Diefenbach, D., Amjad, S., Both, A., Singh, K. & Maret, P. (2017), Trill: A reusable front-end for qa systems, *in* 'ESWC P&D'.

Diefenbach, D., Singh, K., Both, A., Cherix, D., Lange, C. & Auer, S. (2017), The Qanary Ecosystem: getting new insights by composing Question Answering pipelines, *in* 'ICWE'.

Dima, C. (2013), 'Intui2: A prototype system for question answering over linked data', *Proceedings of the Question Answering over Linked Data lab (QALD-3) at CLEF* .

Dima, C. (2014), Answering natural language questions with Intui3, *in* 'Conference and Labs of the Evaluation Forum (CLEF)'.

Dong, L., Wei, F., Zhou, M. & Xu, K. (2015), Question answering over freebase with multi-column convolutional neural networks., *in* 'ACL (1)'.

Dwivedi, S. K. & Singh, V. (2013), 'Research and reviews in question answering system', *Procedia Technology* **10**, 417 – 424.

Fader, A., Soderland, S. & Etzioni, O. (2011), Identifying relations for open information extraction, *in* 'Proceedings of the Conference on Empirical Methods in Natural Language Processing', Association for Computational Linguistics.

Fader, A., Zettlemoyer, L. S. & Etzioni, O. (2013), Paraphrase-driven learning for open question answering., *in* 'ACL (1)', Citeseer.

Ferrandez, O., Spurk, C., Kouylekov, M., Dornescu, I., Ferrandez, S., Negri, M., Izquierdo, R., Tomas, D., Orasan, C., Neumann, G. et al. (2011), 'The QALL-ME framework: A specifiable-domain multilingual question answering architecture', *Web semantics: Science, services and agents on the world wide web* .

Ferré, S. (2013), squall2sparql: a Translator from Controlled English to Full SPARQL 1.1, *in* 'Work. Multilingual Question Answering over Linked Data (QALD-3)'.

Ferré, S. (2017), 'Sparklis: an expressive query builder for sparql endpoints with guidance in natural language', *Semantic Web* **8**(3), 405–418.

Freitas, A. & Curry, E. (2014), Natural language queries over heterogeneous linked data graphs: A distributional-compositional semantics approach, *in* 'Proceedings of the 19th international conference on Intelligent User Interfaces', ACM.

Freitas, A., Curry, E., Oliveira, J. G. & O'Riain, S. (2012), 'Querying Heterogeneous Datasets on the Linked Data Web: Challenges, Approaches, and Trends', *IEEE Internet Computing* .

Freitas, A., Efson Sales, J., Handschuh, S. & Curry, E. (2015), How hard is this query? Measuring the Semantic Complexity of Schema-agnostic Queries, *in* 'Proceedings of the 11th International Conference on Computational Semantics', Association for Computational Linguistics, London, UK.

Gerber, D. & Ngomo, A.-C. N. (2011), Bootstrapping the linked data web, *in* '1st Workshop on Web Scale Knowledge Extraction@ ISWC', Vol. 2011.

Giannone, C., Bellomaria, V. & Basili, R. (2013), 'A HMM-based approach to question answering against linked data', *Proceedings of the Question Answering over Linked Data lab (QALD-3) at CLEF* .

Golub, D. & He, X. (2016), 'Character-level question answering with attention', *arXiv preprint arXiv:1604.00727* .

Google (2016), 'Freebase data dumps', https://developers.google.com/freebase/data.

Hakimov, S., Unger, C., Walter, S. & Cimiano, P. (2015), Applying semantic parsing to question answering over linked data: Addressing the lexical gap, *in* 'Natural Language Processing and Information Systems', Springer.

Hamon, T., Grabar, N., Mougin, F. & Thiessard, F. (2014), Description of the POMELO System for the Task 2 of QALD-2014., *in* 'CLEF (Working Notes)'.

He, S., Zhang, Y., Liu, K. & Zhao, J. (2014), 'CASIA@ V2: A MLN-based Question Answering System over Linked Data', *Proc. of QALD-4* .

Höffner, K. & Lehmann, J. (2015), 'Question Answering on Statistical Linked Data'.

Höffner, K., Walter, S., Marx, E., Usbeck, R., Lehmann, J. & Ngonga Ngomo, A.-C. (2016), 'Survey on Challenges of Question Answering in the Semantic Web', *Semantic Web Journal* .

Jain, S. (2016), Question answering over knowledge base using factual memory networks, *in* 'Proceedings of NAACL-HLT'.

Joris, G. & Ferré, S. (2013), Scalewelis: a scalable query-based faceted search system on top of sparql endpoints, *in* 'Work. Multilingual Question Answering over Linked Data (QALD-3)'.

Kolomiyets, O. & Moens, M.-F. (2011), 'A survey on question answering technology from an information retrieval perspective', *Inf. Sci.* **181**(24), 5412–5434.

Lopez, V., Fernández, M., Motta, E. & Stieler, N. (2012), 'Poweraqua: Supporting Users in Querying and Exploring the Semantic Web', *Semant. web* **3**(3).

Lopez, V., Tommasi, P., Kotoulas, S. & Wu, J. (2016), Queriodali: Question answering over dynamic and linked knowledge graphs, *in* 'International Semantic Web Conference', Springer, pp. 363–382.

Lopez, V., Unger, C., Cimiano, P. & Motta, E. (2013), 'Evaluating question answering over linked data', *Web Semantics: Science, Services and Agents on the World Wide Web* .

Lopez, V., Uren, V., Motta, E. & Pasin, M. (2007), '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.

Lopez, V., Uren, V., Sabou, M. & Motta, E. (2011), 'Is question answering fit for the semantic web? a survey', *Semantic Web* **2**(2).

Lukovnikov, D., Fischer, A., Lehmann, J. & Auer, S. (2017), Neural network-based question answering over knowledge graphs on word and character level, *in* 'Proceedings of the 26th International Conference on World Wide Web', International World Wide Web Conferences Steering Committee, pp. 1211–1220.

Mahendra, R., Wanzare, L., Bernardi, R., Lavelli, A. & Magnini, B. (2011), Acquiring relational patterns from wikipedia: A case study, *in* 'Proc. of the 5th Language and Technology Conference'.

Marginean, A. (2017), 'Question answering over biomedical linked data with grammatical framework', *Semantic Web* **8**(4), 565–580.

Marx, E., Usbeck, R., Ngomo, A.-C. N., Höffner, K., Lehmann, J. & Auer, S. (2014), Towards an open question answering architecture, *in* 'Proceedings of the 10th International Conference on Semantic Systems', ACM.

Mazzeo, G. M. & Zaniolo, C. (2016), 'Answering Controlled Natural Language Questions on RDF Knowledge Bases'.

Nakashole, N., Weikum, G. & Suchanek, F. (2012), PATTY: a taxonomy of relational patterns with semantic types, *in* 'Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning', Association for Computational Linguistics.

Park, S., Shim, H. & Lee, G. G. (2014), ISOFT at QALD-4: Semantic similarity-based question answering system over linked data, *in* 'CLEF'.

Pouran-ebn veyseh, A. (2016), Cross-Lingual Question Answering Using Profile HMM & Unified Semantic Space, *in* 'ESWC'. *to appear*.

Pradel, C., Haemmerlé, O. & Hernandez, N. (2012), A semantic web interface using patterns: the SWIP system, *in* 'Graph Structures for Knowledge Representation and Reasoning', Springer.

Reddy, S., Lapata, M. & Steedman, M. (2014), 'Large-scale semantic parsing without question-answer pairs', *Transactions of the Association for Computational Linguistics* .

Reddy, S., Täckström, O., Collins, M., Kwiatkowski, T., Das, D., Steedman, M. & Lapata, M. (2016), 'Transforming dependency structures to logical forms for semantic parsing', *Transactions of the Association for Computational Linguistics* .

Ruseti, S., Mirea, A., Rebedea, T. & Trausan-Matu, S. (2015), QAnswer-Enhanced Entity Matching for Question Answering over Linked Data, *in* 'CLEF (Working Notes)', CLEF.

Shekarpour, S., Marx, E., Ngomo, A.-C. N. & Auer, S. (2015), 'Sina: Semantic interpretation of user queries for question answering on interlinked data', *Web Semantics: Science, Services and Agents on the World Wide Web* .

Song, D., Schilder, F., Smiley, C., Brew, C., Zielund, T., Bretz, H., Martin, R., Dale, C., Duprey, J., Miller, T. et al. (2015), TR discover: A Natural Language Interface for Querying and Analyzing Interlinked Datasets, *in* 'The Semantic Web-ISWC 2015', Springer.

Ture, F. & Jojic, O. (2016), 'Simple and effective question answering with recurrent neural networks', *arXiv preprint arXiv:1606.05029* .

Unger, C., Bühmann, L., Lehmann, J., Ngonga Ngomo, A.-C., Gerber, D. & Cimiano, P. (2012), Template-based question answering over rdf data, *in* 'Proceedings of the 21st international conference on World Wide Web', ACM, pp. 639–648.

Unger, C., Forascu, C., Lopez, V., Ngomo, A.-C. N., Cabrio, E., Cimiano, P. & Walter, S. (2014), Question answering over linked data (QALD-4), *in* 'Working Notes for CLEF 2014 Conference'.

Unger, C., Forascu, C., Lopez, V., Ngomo, A.-C. N., Cabrio, E., Cimiano, P. & Walter, S. (2015), Answering over Linked Data (QALD-5)., *in* 'Working Notes for CLEF 2015 Conference'.

Unger, C., Ngomo, A.-C. N., Cabrio, E. & Cimiano (2016), 6th Open Challenge on Question Answering over Linked Data (QALD-6), *in* 'The Semantic Web: ESWC 2016 Challenges.'.

Usbeck, R., Ngomo, A.-C. N., Bühmann, L. & Unger, C. (2015), HAWK–Hybrid Question Answering Using Linked Data, *in* 'The Semantic Web. Latest Advances and New Domains', Springer.

Walter, S., Unger, C. & Cimiano, P. (2014), M-ATOLL: a framework for the lexicalization of ontologies in multiple languages, *in* 'The Semantic Web–ISWC 2014', Springer.

Walter, S., Unger, C., Cimiano, P. & Bär, D. (2012), Evaluation of a layered approach to question answering over linked data, *in* 'The Semantic Web–ISWC 2012', Springer.

Wang, Z., Yan, S., Wang, H. & Huang, X. (2014), An overview of microsoft deep qa system on stanford webquestions benchmark, Technical report, Technical report, Microsoft Research.

Wu, F. & Weld, D. S. (2010), Open information extraction using Wikipedia, *in* 'Proceed-

ings of the 48th Annual Meeting of the Association for Computational Linguistics', Association for Computational Linguistics.

Xu, K., Feng, Y. & Zhao, D. (2014), 'Xser@ QALD-4: Answering Natural Language Questions via Phrasal Semantic Parsing'.

Yahya, M., Berberich, K., Elbassuoni, S., Ramanath, M., Tresp, V. & Weikum, G. (2012), Natural language questions for the web of data, *in* 'Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning', Association for Computational Linguistics.

Yahya, M., Berberich, K., Elbassuoni, S. & Weikum, G. (2013), Robust question answering over the web of linked data, *in* 'Proceedings of the 22nd ACM international conference on Conference on information & knowledge management', ACM.

Yang, M.-C., Duan, N., Zhou, M. & Rim, H.-C. (2014), Joint relational embeddings for knowledge-based question answering., *in* 'EMNLP'.

Yang, M.-C., Lee, D.-G., Park, S.-Y. & Rim, H.-C. (2015), 'Knowledge-based question answering using the semantic embedding space', *Expert Systems with Applications* .

Yao, X. (2015), Lean question answering over freebase from scratch., *in* 'HLT-NAACL'.

Yao, X. & Van Durme, B. (2014), Information Extraction over Structured Data: Question Answering with Freebase., *in* 'ACL (1)', Citeseer.

Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M. & Soderland, S. (2007), Textrunner: open information extraction on the web, *in* 'Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations', Association for Computational Linguistics.

Yavuz, S., Gur, I., Su, Y., Srivatsa, M. & Yan, X. (2016), Improving semantic parsing via answer type inference., *in* 'EMNLP', pp. 149–159.

Yih, S. W.-t., Chang, M.-W., He, X. & Gao, J. (2015), 'Semantic parsing via staged query graph generation: Question answering with knowledge base'.

Yih, W.-t., Richardson, M., Meek, C., Chang, M.-W. & Suh, J. (2016), The value of semantic parse labeling for knowledge base question answering., *in* 'ACL (2)'.

Yin, W., Yu, M., Xiang, B., Zhou, B. & Schütze, H. (2016), 'Simple question answering by attentive convolutional neural network', *arXiv preprint arXiv:1606.03391* .

Yosef, M. A., Hoffart, J., Bordino, I., Spaniol, M. & Weikum, G. (2011), 'Aida: An online tool for accurate disambiguation of named entities in text and tables', *Proceedings of the VLDB Endowment* **4**.

Zettlemoyer, L. S. & Collins, M. (2012), 'Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars', *arXiv preprint arXiv:1207.1420* .

Zhang, Y., Liu, K., He, S., Ji, G., Liu, Z., Wu, H. & Zhao, J. (2016), 'Question answering over knowledge base with neural attention combining global knowledge information', *arXiv preprint arXiv:1606.00979* .

Zhange, Y., He, S., Liu, K. & Zhao, J. (2016), 'A Joint Model for Question Answering over Multiple Knowledge Bases'.

Zhu, C., Ren, K., Liu, X., Wang, H., Tian, Y. & Yu, Y. (2015), 'A Graph Traversal Based Approach to Answer Non-Aggregation Questions Over DBpedia', *arXiv preprint arXiv:1510.04780* .

Zou, L., Huang, R., Wang, H., Yu, J. X., He, W. & Zhao, D. (2014), Natural language question answering over RDF: a graph data driven approach, *in* 'Proceedings of the 2014 ACM SIGMOD international conference on Management of data', ACM.

Author Biographies

Dennis Diefenbach is a european resercher. In 2010 he received a pre-degree in physics. He got a B.S. and a M.S. in mathematics in 2012 and 2014 respectively, at the University of Kaiserslautern, Germany. The specialization was in algebra, geometry and computer algebra. In 2015 he received a Marie Sklodowska-Curie fellowship in the framework of the WDAqua project (`www.wdaqua.eu`). He is a Ph.D. Student at Laboratoire Hubert Curien in Saint-Etienne, France. His research interest is in Question Answering over Knowledge Bases and related topics. He is the main developer behind a reusable front-end for QA systems called Trill `https://github.com/WDAqua/Trill` and the QA system called WDAqua-core0 that can be found under `www.wdaqua.eu/qa`.

Vanessa Lopez is a researcher at IBM Research Ireland since 2012, where she investigates solutions for harnessing urban and web data as city knowledge, through Linked Data technologies to support data integration, and envision natural ways for users to query, explore and find useful insights across data sources. Her research has been applied to develop context-aware applications for smarter cities and Social and Health care. Previous to joining IBM, she was a researcher at KMi (Open University) from 2003, where she investigated NL Question Answering interfaces for the Web of Data and received a PhD degree. She graduated in 2002 with a degree in computer engineer from the Technical University of Madrid (UPM), where she held an internship at the AI Lab. She has co-authored more than 40 publications in high impact conference and journals.

Kamal Deep Singh received the B.Tech. degree in electrical engineering from the Indian Institute of Technology Delhi (IITD), Delhi, India, in 2002 and the Ph.D. degree in computer science from the University of Rennes 1, Rennes, France in 2007. He then joined the Dionysos Group, National Research Institute in Computer Science (INRIA), as a Postdoctoral Researcher, where he codeveloped many components of quality-of-experience estimation tools and worked mainly on the analysis of video-based applications. After that, he was a Postdoctoral Researcher with Telecom Bretagne, Rennes, where he worked on Internet of Things and cognitive radio. He is currently an Associate Professor with the Université of Saint-Étienne, Saint-Étienne, France, where he is part of the research team called Connected Intelligence at the Laboratoire Hubert Curien. His research interests include Internet of Things, smart cities, big data, semantic web, quality of experience, and software-defined networking.

Pierre Maret is a Professor in Computer Science at the University Jean Monnet (University of Lyon), Laboratoire Hubert Curien, in Saint Etienne since 2009. He received a PhD in Computer Science in 1995 and became an Associate Professor at INSA Lyon in 1997. His research interest are data and knowledge modeling, semantic web, knowledge management, social networks, virtual communities. He conducts research in collaboration with international research groups and industry. He leads the ITN Marie Sklodowska-Curie WDAqua for its French part, and co-chairs the workshop Web Intelligence and Communities hosted at The Web Conference (W3c). He leads the international master track Cyber-Physical and Social Systems (CPS2) and coordinates the international relations of the Faculty of Sciences and Technologies.

Correspondence and offprint requests to: Dennis Diefenbach, Université de Lyon, CNRS UMR 5516 Laboratoire Hubert Curien, F-42023, Saint-Etienne, France. Email: dennis.diefenbach@univ-st-etienne.fr