Towards an Open Question Answering Architecture

Edgard Marx
AKSW/BIS, Universität Leipzig
PO Box 100920, 04109
Leipzig, Germany
marx@informatik.uni-leipzig.de

Ricardo Usbeck
Universität Leipzig, IFI/AKSW
Unister GmbH, Leipzig
usbeck@informatik.uni-leipzig.de

Axel-Cyrille Ngonga Ngomo
AKSW/BIS, Universität Leipzig
PO Box 100920, 04109
Leipzig, Germany
ngomo@informatik.uni-leipzig.de

Konrad Höffner
AKSW/BIS, Universität Leipzig
PO Box 100920, 04109
Leipzig, Germany
hoffner@informatik.uni-leipzig.de

Jens Lehmann
AKSW/BIS, Universität Leipzig
PO Box 100920, 04109
Leipzig, Germany
lehmann@informatik.uni-leipzig.de

Sören Auer
CS/EIS, Universität Bonn
Römerstraße 164, 53117
Bonn, Germany
auer@cs.uni-bonn.de

ABSTRACT

Billions of facts pertaining to a multitude of domains are now available on the Web as RDF data. However, accessing this data is still a difficult endeavour for non-expert users. In order to meliorate the access to this data, approaches imposing minimal hurdles to their users are required. Although many question answering systems over Linked Data have being proposed, retrieving the desired data is still significantly challenging. In addition, developing and evaluating question answering systems remains a very complex task. To overcome these obstacles, we present a modular and extensible open-source question answering framework. We demonstrate how the framework can be used by integrating two state-of-the-art question answering systems. As a result our evaluation shows that overall better results can be achieved by the use of combination rather than individual stand-alone versions.

1. INTRODUCTION

Since its inception, the Web has been used by millions of users with the aim of sharing, reproducing and linking information. More and more structured data, e.g., using the Resource Description Framework (RDF), is made available by users, services and more recently sensors. Consequently, machines are now empowered to use a large amount of data available on the Web. Examples of such datasets are *DBpedia* [6][1] and *LinkedGeoData* [10][2], which encompass more than 1 billion triples each. The main advantage of these datasets is that they represent a variety of knowledge across several domains. For example, *astronauts who took part in the Apollo 14 mission* can be easily retrieved from DBpedia. Although this data is accessible, users still face difficulties when trying to retrieve the desired information using traditional methods.

In particular, state-of-the-art search engines fail to retrieve the set of resources intended by the user because most of them abide by the *bag of words* paradigm. For instance, the query *Apollo 14 astronauts* on search engines such as Sigma[3] and Sindice[4] will retrieve relevant documents rather than the desired set of resources. Furthermore, in some engines fail finding a suitable answer, due to no full understanding of the input query or not properly indexed data in the underlying knowledge base.

Over the last years, several approaches for question answering (QA) based on the Web of Data have been proposed. However, retrieving the desired data is still significantly challenging. Reported results from the Question Answering on Linked Data (QALD) benchmark[5] indicate that only $\approx 54\%$ of the given questions were correctly answered in QALD-1 and 32% in the more difficult QALD-2 and QALD-3 benchmarks.

In this paper, we present *openQA*, an open source question answering framework that combines several question answering systems. The proposed framework can facilitate the evaluation by promoting the modularization and easy integration of new modules via a plug-in architecture. Moreover, our experiments reveal a considerable improvement of correct answers by combining question answering systems, i.e., 87.5% using the QALD-3 benchmark.

The remainder of this paper is structured as follows. Section 2 presents an background overview and related work. Section 3 introduces the framework architecture and its components. Section 4 details the implemented components and used technologies. Section 5 presents the achieved results. Finally, section 6 concludes with an outlook on future work.

[1]`http://dbpedia.org`, version 3.9

[2]`http://linkedgeodata.org`, version of March 23rd, 2014

[3]`http://sig.ma`

[4]`http://sindice.com`

[5]`http://greententacle.techfak.uni-bielefeld.de/~cunger/qald`

SEM '14 September 04 - 05 2014, Leipzig, AA, Germany
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2927-9/14/09 ...$15.00.
http://dx.doi.org/10.1145/2660517.2660519

2. BACKGROUND AND RELATED WORK

The World Wide Web Consortium recommend the use of Linked Data (LD) standard to represent and publish open data[6]. The LD standard uses for the representation of data (facts, rules and their relationships) the *RDF* format.

Question answering on linked data is an active and diverse research field with many existing research prototypes designed for different environments. Systems running on a *closed domain* are optimized for and only work on a specific knowledge base or field, like biology [1] or medicine [2]. Since they do not need to integrate different schemas and knowledge bases and thus suffer less from ambiguity, they generally create faster and higher-quality results. However, closed-domain approaches lack flexibility. Additionally, there are high costs when adapting such a system to a new domain or implementing a new system. Thus, the research focus has been moved towards multi-purpose, open-domain QA systems like FREyA [4] or hybrid approaches like TBSL [11] with a domain unspecific core and a domain specific, adaptable extension.

Question answering is a complex process which consists of many different steps, most of which have to be executed in a sequential manner and are thus called a pipeline. Some of those steps, like natural language processing (NLP), which involves parsing and part-of-speech extraction can make use of mature, well performing tools. No such simple solution exists however for the step of interpretation – the process of extracting the meaning of a question.

FREyA [4] is different from SPARQL driven systems and makes use of a combination of syntactic parse trees, phrases potentially representing ontologies as well as ontology concepts. Phrases potentially representing ontology concepts, so-called *Potential Ontology Concepts (POCs)* are identified by heuristics on the syntactic parse tree. *Ontology Concepts (OCs)*, are resources from an ontology that are identified matching their labels with phrases from the question without regarding its structure. A consolidation algorithm then matches POCs to OCs. In case of ambiguities, feedback from the user is asked. Disambiguation candidates are created using string similarity in combination with WordNet synonym detection. The system learns from the user selections, thereby improving the precision over time. However, user feedback is rare and expensive.

Most frequently, QALD systems uses a set of predefined graph pattern templates in the interpretation process to generate SPARQL queries representing the input question. The use of SPARQL queries enables the representation of a more complex semantic structure of the original natural language question, going beyond the limitation of triple-only-based systems. However, the strategy used to generate the graph patterns and consequently the SPARQL query differs significantly between approaches. This is the case of *SINA* [9], *TBSL* [11] and *Casia* [5].

SINA is a question answering system that uses the knowledge base itself to formulate the graph pattern templates. The system deals with the input query as being a set of keywords, which enable the interpretation of keyword as well as full question fashion queries. In order to prune the number of generated graph pattern templates, *SINA* uses a statistical Hidden Markov model.

TBSL is another template-based question answering system for LD. Different from other approaches, *TBSL* uses an entity identification and predicate detection method to fill the slots of the previosuly defined SPARQL template query with data extracted from the original query.

Casia [5] generates the graph pattern templates by using the question type, named entities and POS tags techniques. The generated graph patterns are then mapped to resources using Wordnet, PATTY and similarity measures. Finally, the possible graph pattern combinations are used to build SPARQL queries.

As each of these systems has its strengths and weaknesses, we move a step further, adopting a composed solution.

3. FRAMEWORK DESIGN

We propose a framework for combining different approaches to question answering. The design was made taking into account the architecture of classic and over linked data QA systems, search engines, smart assistants as well as hybrid systems such as IBM Watson [7]. In the following, we explain our *openQA* framework architecture which consists of four main modules comprising several sub modules, see Figure 1. The framework has a main work-flow comprising four stages (*interpretation*, *retrieval*, *synthesis*, *renderization*) and adjacent modules (*context* and *service*). The adjacent modules are intended to be accessed by any of the components of the main work-flow.

One of the biggest challenges in realizing a generic framework is how to combine different approaches. To tackle this challenge, the framework was designed in a plug-in architecture style. Plug-in architectures are known for promoting modularization and easy integration. We aim that the openQA architecture allows to support the diversity of many existing Linked Data question answering systems. *openQA* is organized as follows:

Answer Formulation. The core module comprises the *answer formulation* process, its purpose is to receive an input query and deliver the most probable answer candidates. The answer formulation process is composed by four different stages:

1. *Interpretation* The first and crucial step of the core module is the *interpretation*. Here, the framework attempts to generate a formal representation of the (intention behind the) input question. By these means, it also determines how the input question will be processed by the rest of the system. Because the interpretation process is not trivial, there is a wide variety of techniques that can be applied at this stage, such as tokenization, disambiguation, internationalization, logical forms, semantic role labels, question reformulation, coreference resolution, relation extraction and named entity recognition amongst others. Most of these technologies are well understood and will not be discussed here. The interpretation stage can generate one or more interpretations of the same input in different formats such as SPARQL, SQL or string tokens.

2. *Retrieval* After an interpretation of the given question is generated, the *retrieval* stage extracts answers from sources according to the delivered interpretation format or content. For instance, one of the outputs of the interpretation can be: (1) a SPARQL query which can be handled by a triple store; (2) an SQL query which will be processed by a relational database,

[6]`http://www.w3.org/standards/semanticweb`

[7]`http://www.ibm.com/smarterplanet/us/en/ibmwatson`

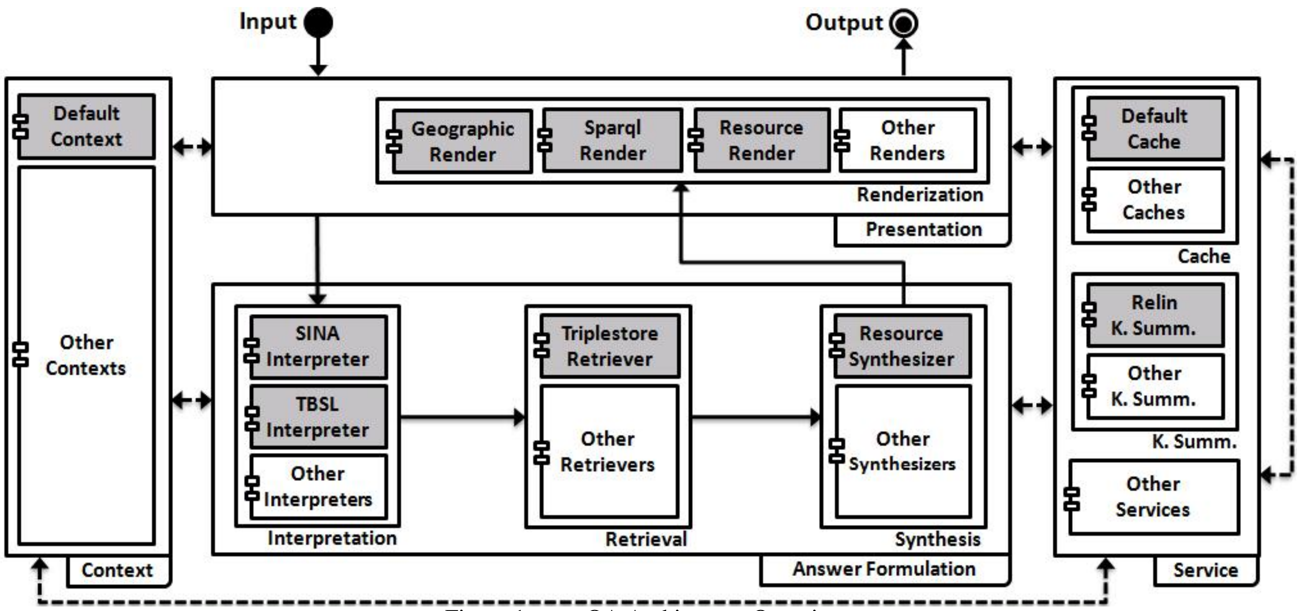


Figure 1: openQA Architecture Overview.

or; (3) a set of keywords that can be sent to a document-based search engine. Specific interpretations can also be used for extracting answers from sources such as web services.

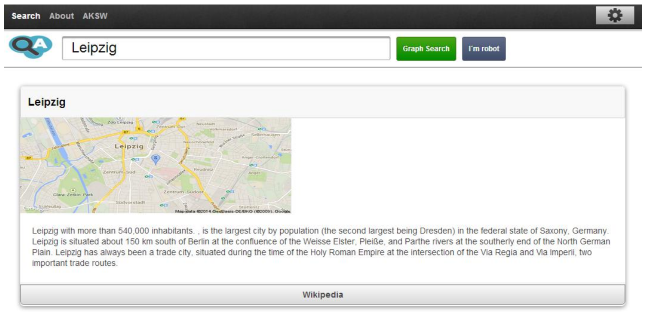
3. *Synthesis* Answers may be extracted from different sources, be ambiguous and redundant. To alleviate this issues the *Synthesis* stage processes all information from different retrieval sub-modules. Results that appear multiple times are fused with an occurrence attribute that helps to rank, cluster and estimate the confidence of the retrieved answer candidates. The *Synthesis* process is closed by *Definition 1*.

DEFINITION 1 (SYNTHESIS). *Let $R_1, \ldots, R_n$ be the result sets generated by different search. The result of the synthesis process is a duplicate-free result set $R = \bigcup_{i=1 \ldots n} R_i$. The ranking of the results in R is left open to the user but must be specified by the means of an injective function pos which maps each $r \in R$ to exactly one value between 1 and $|R|$.*

Renderization. Typically a knowledge base comprises data in several types and formats. Each type of information demands a specific display style to deliver the best possible amount of information to the user. In the proposed architecture the concepts of answer *formulation* and *presentation* are detached. The *renderization* module encapsulates the answer presentation and comprises one or more renders that work as widgets. That is, each widget plugged to the *renderization* module can be used to render one type or format of data.

Context. Users context can help the answer personalization providing information such as who or where is the user, what is the preferred language or if the question is one of a series of related previous questions. To enable the system to produce better answers the *context* module can store users information, such as location, statistics and previous queries.

Services. The *services* modules are designed to be easily accessed by the components. They intend to facilitate the encapsulation, addition and sharing of application functionalities between the modules i.e.:



Leipzig with more than 540,000 inhabitants. , is the largest city by population (the second largest being Dresden) in the federal state of Saxony, Germany. Leipzig is situated about 150 km south of Berlin at the confluence of the Weisse Elster, Pleiße, and Parthe rivers at the southerly end of the North German Plain. Leipzig has always been a trade city, situated during the time of the Holy Roman Empire at the intersection of the Via Regia and Via Imperii, two important trade routes.

Figure 2: Screenshot of the demo available at `http://openqa.aksw.org`.

1. *Cache* The result of a determined parameter in a function can be required multiple times. The *cache* service intends to provide efficiency while prevents the execution of the same process multiple times. It stores the results of processes in order that future requests to the same process can be executed faster.

2. *Knowledge Summarization* Currently, the *DBpedia*[8] dataset comprises millions of facts distributed over millions of entities, but not all this data is useful for users. *Knowledge Summarization* has an important role in knowledge presentation. Knowledge summarization refers to a combination of two techniques, set and entity summarization. Set and entity summarization consist of the use of heuristics to rank respectively entities and their attributes.

4. IMPLEMENTATION

The framework is implemented as open-source and additionally provides several built-in components and information about: (1) possible log messages, (2) used memory, (3) errors as well as (4) answered and not answered queries. Figure 1 gives an overview of the implemented modules in gray. Moreover, a test integration framework was developed in order to run evaluation queries on *openQA* and the underlying modules.

[8]`http://wiki.dbpedia.org/About`

Combining Approaches. Regarding *Knowledge Summarization*, an algorithm based on *Relin* [3] has been added. *SINA* and *TBSL* were combined as interpretation modules. The framework can handle SPARQL queries over a retrieval component based on the *Jena*[9] client. The synthesis module is inspired by [7] and formalized by Definition 1 in which $pos(r) = -\log(|r|/|R|)$, i.e., the more occurrences an answer has, the more confidence it provides. The renderization module of *openQA* uses *Infograph*. *Infograph* is a render API developed for *openQA* that can render (1) SPARQL queries, (2) literals and (3) entities from knowledge bases. Infograph is capable of differentiating entities with and without geographic information as well as automatically generate entity descriptions based on their attributes via *SPARQL2NL* [8], in case the entity does not have a description.

5. EVALUATION

The evaluation of the described framework was performed measuring improvement of correct answered queries in QALD-3 by combining the participant systems. Furthermore, the framework was assessed with the fourth version of the same benchmark (QALD-4), by using stand-alone versions of *TBSL* and *SINA* as well as a combination of both. All evaluations were executed over the multilingual tasks. The correctness and incorrectness synthesized answer from different systems is given by logical conjunction. According to QALD benchmark the result from the synthesis operations should be equal to the given correct answer set C otherwise is considered incorrect.

$$\forall x \; C(x) \iff \bigvee_{i=1}^{n} R_i(x)$$

The QALD-3 evaluation (Figure 3) shows that a significant improvement of correct answers can be achieved (87.5%) by using a theoretical combination of all the participant systems.

Regarding QALD-4, we measure a combination of the current implemented versions of *SINA* and *TBSL* (trained respectively using QALD-2 and QALD-1). As both systems were still under major revision, the tests were focused in how much improvement could be achieved under their combination. The results reveal that a combination of both systems is also better than a stand-alone versions, leading to an improvement of 11% in correct answered queries.

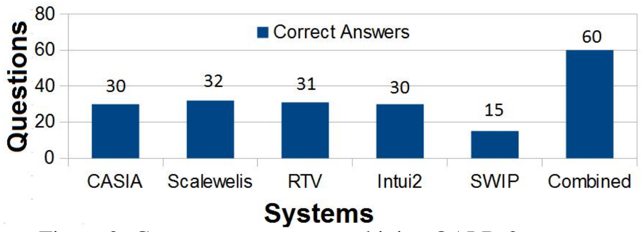


Figure 3: Correct answers on combining QALD-3 systems.

The framework (Figure 2) is available over two public instances, one using *SINA* (`http://sina.aksw.org`) and another using both, *SINA* and *TBSL* (`http://openqa.aksw.org`). As the live instances rely on the public *DBpedia* endpoint, the user's experience can be affected by instabilities. Regarding this, a demo video is also available at `http://www.aksw.org/Projects/openQA`.

[9]`http://jena.apache.org`

6. CONCLUSIONS AND FUTURE WORK

The proposed architecture is part of a larger research agenda which aims to define and develop a common framework for question answering systems. Therefore, we have implemented several components, which are open source and can be downloaded, plugged and extended [10]. The achieved results indicate that the *openQA* architecture can be used as a base for developing, combining and evaluating question answering systems. The next efforts will consist of integrating unstructured sources, such as text extracted data from web pages and document bases. Furthermore, we want to extend the implemented modules with non-commercial APIs from other question answering systems and evaluate them against several well-know benchmarks.

Acknowledgments
This work has been supported by the ESF and the Free State of Saxony and the FP7 project GeoKnow (GA No. 318159) and by CNPq, under the program Ciências Sem Fronteiras.

7. REFERENCES

[1] S. J. Athenikos and H. Han. Biomedical question answering: A survey. *Computer methods and programs in biomedicine*, 99(1):1–24, 2010.

[2] A. Ben Abacha and P. Zweigenbaum. Medical question answering: translating medical questions into sparql queries. In *Proc. of the 2nd ACM SIGHIT International Health Informatics Symposium*, pages 41–50. ACM, 2012.

[3] G. Cheng, T. Tran, and Y. Qu. Relin: relatedness and informativeness-based centrality for entity summarization. In *The Semantic Web–ISWC 2011*, pages 114–129. Springer, 2011.

[4] D. Damljanovic, M. Agatonovic, and H. Cunningham. Freya: An interactive way of querying linked data using natural language. In *The Semantic Web: ESWC 2011 Workshops*, pages 125–138. Springer, 2012.

[5] S. He, S. Liu, Y. Chen, G. Zhou, K. Liu, and J. Zhao. Casia@qald-3:a question answering system over linked data. CLEF, 2013.

[6] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 2014.

[7] V. Lopez, A. Nikolov, M. Fernández, M. Sabou, V. S. Uren, and E. Motta. Merging and ranking answers in the semantic web: The wisdom of crowds. In A. Gómez-Pérez, Y. Yu, and Y. Ding, editors, *ASWC*, volume 5926 of *Lecture Notes in Computer Science*, pages 135–152. Springer, 2009.

[8] A.-C. Ngonga Ngomo, L. Bühmann, C. Unger, J. Lehmann, and D. Gerber. Sparql2nl - verbalizing sparql queries. In *Proc. of WWW 2013 Demos*, pages 329–332, 2013.

[9] S. Shekarpour, A.-C. N. Ngomo, and S. Auer. Question answering on interlinked data. In *WWW*, pages 1145–1156, 2013.

[10] C. Stadler, J. Lehmann, K. Höffner, and S. Auer. Linkedgeodata: A core for a web of spatial open data. *Semantic Web Journal*, 2011.

[11] C. Unger, L. Bühmann, J. Lehmann, A.-C. N. Ngomo, D. Gerber, and P. Cimiano. Template-based question answering over rdf data. In *21st international conference on World Wide Web*, pages 639–648, 2012.

[10]`http://aksw.org/Projects/openQA`