An Open Question Answering Framework

Edgard Marx, Tommaso Soru, Diego Esteves, Axel-Cyrille Ngonga Ngomo, and Jens Lehmann

AKSW, Institute of Computer Science, University of Leipzig
http://openqa.aksw.org

Abstract. Question answering systems are currently regarded as one of the key technologies to empower lay users to access the Web of Data. Nevertheless, they still pose hard and yet unsolved research challenges. Additionally, developing and evaluating question answering systems to address those challenges is a very complex task. We present openQA, a modular open-source platform that allows developing and evaluating question answering systems. We show the different interfaces implemented by openQA as well as the different processing steps from a question to the corresponding answer. We demonstrate how openQA can be used to implement, integrate, evaluate and instantiate question answering systems easily by combining two state-of-the-art question answering approaches.

1 Introduction

The use of Semantic Web technologies led to an increase of machine-readable data published on the Web. Examples of datasets are DBpedia[1][1] and LinkedGeoData[2][4], which encompass more than 1 billion facts each. However, retrieving the desired information from these immense sources of knowledge is still challenging. Over the last few years, several approaches for question answering (QA) over the Web of Data have been proposed to address this problem. In this article, we demonstrate openQA [2], an open source question answering platform that enables the reconciliation of different QA architectures and approaches. The aim of openQA is to provide a common platform for developing, testing and instantiating QA systems.

2 Demonstration

The goal of the demonstration will be to show how to integrate, evaluate and instantiate a QA application using the openQA platform. In addition, we will show how openQA implements different processing steps from a natural-language (NL) query to the corresponding answer. In the demonstration, we also go over some of the implemented plug-ins and applications, in particular SINA [3] and TBSL [5]. At the end, we will discuss the benefits and disadvantages of the platform as well as how we plan to address them in the future. In the following, we detail the core parts of the framework that will be explicated in the demo.

[1] http://dbpedia.org
[2] http://linkedgeodata.org

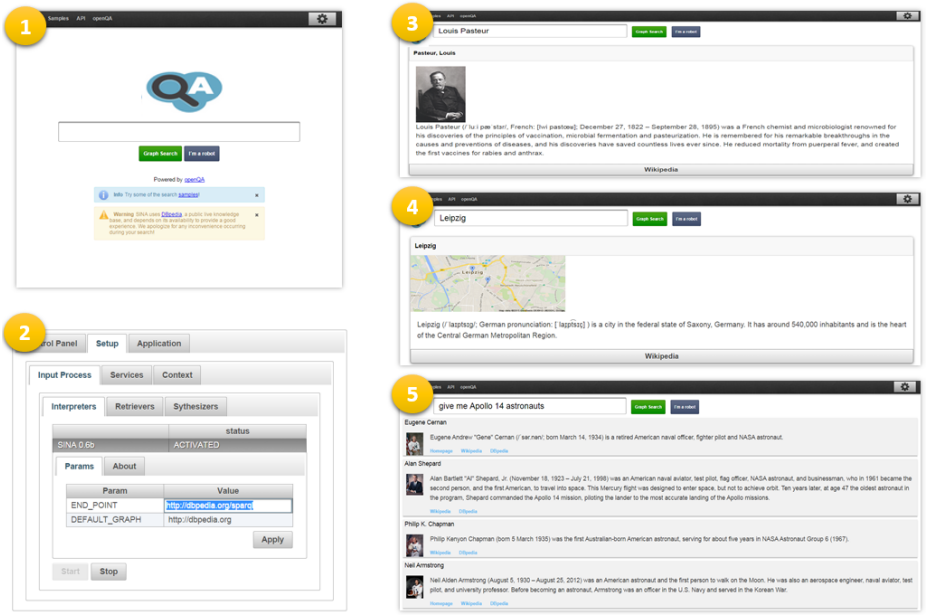

Fig. 1. `openQA` Web server overview. An overview of five different features available in the `openQA` Web server: (1) Welcome Search page; (2) Management Panel: the management panel enables the user to change the status of plug-ins (activated, deactivated), check logs and customize parameters; Result page: The `openQA` Web server can render three different type of answers: (3) Literal/Entity, (4) Geo-Entity and (5) Literal/Entity Set.

2.1 The Answer Formulation Workflow

The main workflow of the `openQA` framework is implemented in the *Answer Formulation* module. It encompasses the process of formulating an answer from a given input (see (1) in Figure 1) query and comprises three 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, `openQA` 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. 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. Specific interpretations can also be used for extracting answers from sources such as web services.

3. *Synthesis:* Answers may be extracted from different sources. Thus, they might be ambiguous and redundant. To alleviate these 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 (see (3)-(5) in Figure 1).

In addition to to the *Answer Formulation*, there are two other modules: the (1) *Service* and the (2) *Context*. The *Context* module allows a more personalized answer and contains information such as users location, statistics, preferences and previous queries. It is useful for instance, to rank or determine the language of the answer, e.g., in a system with support for internationalization.[3]. Thus, the same query can generate different answers in different contexts. The *Services* modules are designed to easily add, extend and share common features among the modules, e.g., a common cache structure. Both modules are accessible by any of the components in the main workflow via *dependency injection*.

We will explain the workflow of openQA platform using examples.[4]

2.2 Combining And Evaluating

An analysis of a theoretical combination of all the participant systems in Question Answering Over Linked Data 3 (QALD-3) shows that the number of correct answers can be improved by 87.5% [2]. Moreover, we implement two different interpreter modules using SINA and TBSL. Our evaluation shows that the combination of the two interpreters leads to an improvement of 11% of correctly answered queries in QALD-4 when compared with their stand-alone versions.

The* openQA *test suite There are different aspects in evaluating question answering systems, e.g., runtime, accuracy and usability. Regarding the accuracy, there are various benchmarks i.e., SemanticSerch'10[5] and QALD.[6] SemanticSearch'10 is based on users queries extracted from YAHOO search log, with an average distribution of 2.2 words per-query. QALD is designed by experts and focuses on accurately producing an answer and the generation of a formal representation of it in the form of a SPARQL query. Therefore, there are more elaborated queries in QALD compared to SemanticSearch'10. The `openQA` platform implements a built-in test suite to evaluate question answering systems using QALD benchmarks. The test suite enables users to measure the accuracy and runtime performance of the system. Furthermore, to facilitate debugging and development, it is also possible to trace the stack of each entry in the generated answer.

[3] `http://www.w3.org/standards/webdesign/i18n`
[4] `http://openqa.aksw.org/examples.xhtml`
[5] `http://km.aifb.kit.edu/ws/semsearch10/`
[6] `http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/`

We will show how to implement, integrate and evaluate modules in the `openQA` platform using examples as well as some of the available `plug-ins`.

2.3 Instantiating

The `openQA` platform implements a `Web server` that enables an easy instantiation of QA applications (Figure 1). Two examples of such applications using `openQA` are SINA, available at `http://sina.aksw.org`, and the `openQA` demo. The `openQA` demo combines several openQA plug-ins, e.g., the SINA and TBSL interpreters. It is accessible at `http://search.openqa.aksw.org`. As the live demo instances use the public *DBpedia* endpoint, the user's experience can be affected by instabilities. Thus, a demo video is also available at `http://openqa.aksw.org`. The `Web server` can display the resulting answer materialized in a web page or as a formal SPARQL query. It also implements a customized search and a plug-in management interface as well as a built-in `RESTful interface`. The `REST API` serializes the result in JSON format.

We will show how to instantiate and manage the `openQA Web server` (see (2) in Figure 1) using the available instances.

3 Conclusion

During the demo, we will present an open-source and extensible framework that can be used to implement, integrate, evaluate and instantiate question answering systems easily. The presented work is a part of a larger agenda to define and develop a common platform for question answering systems. The next efforts will consist of (1) the integration of approaches targeting unstructured data, (2) facilitating the deployment of plug-ins as well as (3) the workflow instantiation. Furthermore, we want to extend the implemented show cases and the number of the systems integrated in the `openQA` platform.

References

1. J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. 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*, 5:1–29, 2015.
2. E. Marx, R. Usbeck, A.-C. N. Ngomo, K. Höffner, J. Lehmann, and S. Auer. Towards an open question answering architecture. In *ICSC*, SEM '14, pages 57–60, New York, NY, USA, 2014. ACM.
3. S. Shekarpour, E. Marx, A.-C. N. Ngomo, and S. Auer. Sina: Semantic interpretation of user queries for question answering on interlinked data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 30(0):39 – 51, 2015. Semantic Search.
4. C. Stadler, J. Lehmann, K. Höffner, and S. Auer. LinkedGeoData: A core for a web of spatial open data. *Semantic Web Journal*, 3(4):333–354, 2012.
5. 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.