Entailment-based Question Answering for Structured Data

Bogdan Sacaleanu♣, Constantin Orasan♦, Christian Spurk♣, Shiyan Ou♦, Oscar Ferrandez♥, Milen Kouylekov♠ and Matteo Negri♠
♣LT-Lab, DFKI GmbH / Saarbrücken, Germany
♦RIILP, University of Wolverhampton / Wolverhampton, UK
♠Fondazione Bruno Kessler (FBK) / Trento, Italy
♥University of Alicante / Alicante, Spain

Abstract

This paper describes a Question Answering system which retrieves answers from structured data regarding cinemas and movies. The system represents the first prototype of a multilingual and multimodal QA system for the domain of tourism. Based on specially designed domain ontology and using Textual Entailment as a means for semantic inference, the system can be used in both monolingual and cross-language settings with slight adjustments for new input languages.

1 Introduction

Question Answering over structured data has been traditionally addressed through a deep analysis of the question in order to reconstruct a logical form, which is then translated in the query language of the target data (Androutsopoulos et al, 1995, Popescu et al, 2003). This approach implies a complex mapping between linguistic objects (e.g. lexical items, syntactic structures) and against data objects (e.g. concepts and relations in a knowledge base). Unfortunately, such a mapping requires extensive manual work, which in many cases represents a bottleneck preventing the realization of large scale and portable natural language interfaces to structured data.

This paper presents the first prototype of a question answering system which can answer questions in several languages about movies and cinema using a multilingual ontology and textual entailment. The remainder of the paper is structured as follows: Section 2 presents the concept of entailment-based question answering; Section 3 describes our prototype which implements this concept; A brief evaluation is presented in Section 4, followed by conclusions in Section 5.

2 Entailment-based QA

Recently Textual Entailment (TE) has been proposed as a unifying framework for applied semantics (Dagan and Glickman, 2004), where the need for an explicit representation of a mapping between linguistic objects and data objects can be, at least partially, bypassed through the definition of semantic inferences at a textual level. In this framework, a text (T) is said to entail a hypothesis (H) if the meaning of H can be derived from the meaning of T.

On the basis of the TE framework, the QA problem can be recast as an entailment problem, where the text (*T*) is the question (or its affirmative version) and the hypothesis (*H*) is a relational answer pattern, which is associated to instructions for retrieving the answer to the input question. In this framework, given a question *Q* and a set of relational answer patterns *P*, a QA system needs to select those patterns in *P* that are entailed by *Q*. Instructions associated to answer patterns may be viewed as high precision procedures for answer extraction, which are dependent on the specific source which is asked for. In case of QA over structured data, instructions could be queries to a database; whilst in case of QA on the Web, an instruction could be the URL of a Web page containing the answer to a question or some form of IR query to a search engine.

Therefore, the underlying idea of an entailment-based QA system is to match the user's request to a set of predefined question patterns in order to get some kind of analysis for the request.

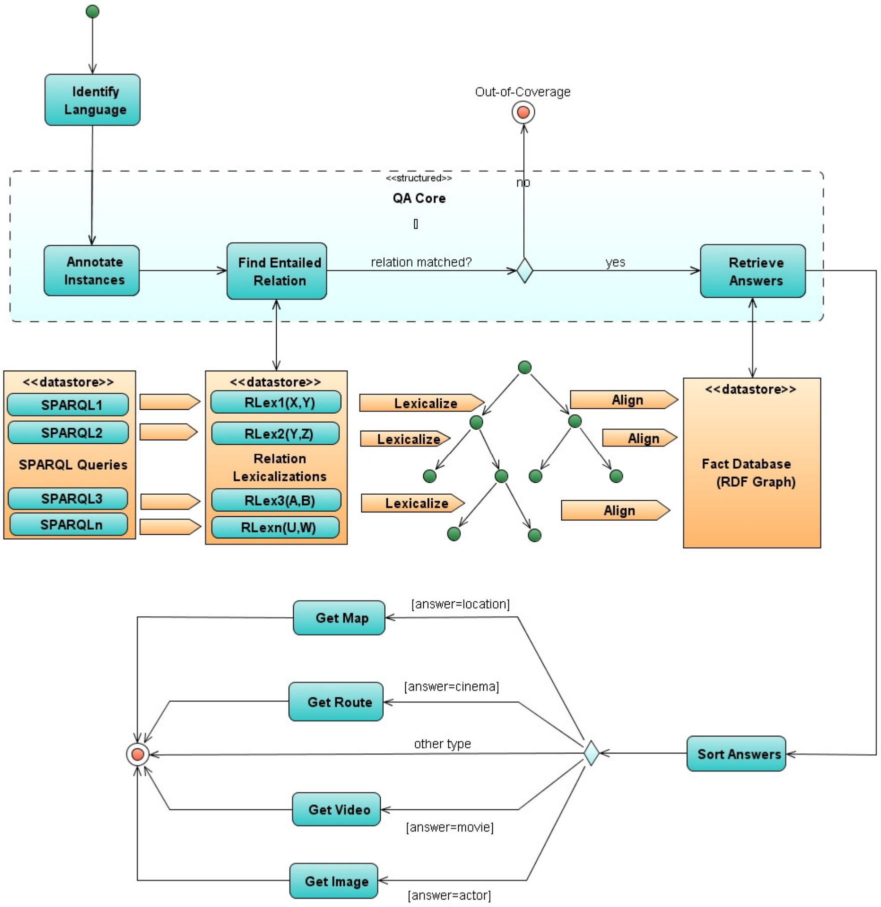

Figure 1. System Architecture

As an example consider the question "*Where can I watch the movie 'Dreamgirls' next Saturday?*" and the predefined question patterns:

- Which movies are currently running in [CINEMA]? → EAT = [MOVIE]
- Where can I watch the movie [MOVIE] on [WEEKDAY]? → EAT = [CINEMA]
- Where can I see [MOVIE]? → EAT = [CINEMA]

In the example, each of the patterns contains placeholders for relevant named entities and has an *expected answer type* (EAT) associated with it. The entailment-based QA system should return that pattern (2) is entailed by the question and as a result the retrieval instructions associated to it will be used to answer the question.

3 Description of system

Our question answering system implements the concept of entailment-based question answering described in the previous section. The overall structure of our system is presented in Figure 1.

Given a question asked by a user of the system in a known location, the QA planner forwards it to the *Instance Annotator* in order to find any concepts that might be related to the targeted domain (i.e. cinema, city, movie). The result is then analyzed by the *Relation Matcher*, which on the basis of entailment can either select the most appropriate interpretation of the question and implicitly its associated procedure of answering the question, or decide that the user request is **out-of-coverage** if no such interpretation is available.

The cross-linguality of our system and, to a certain extent, the interaction between its components is ensured by a domain ontology which is used for all four languages involved in the project: English, German, Italian and Spanish, and its modules (Ou et al., 2008). Concepts from the ontology are used to annotate the user questions as well as data from which the answer is extracted. In the current stage of the project, the answers are contained in databases obtained from content provides or built from structured web pages. As a result, the information in the database tables was annotated with concepts from the ontology and then converted into an RDF graph to

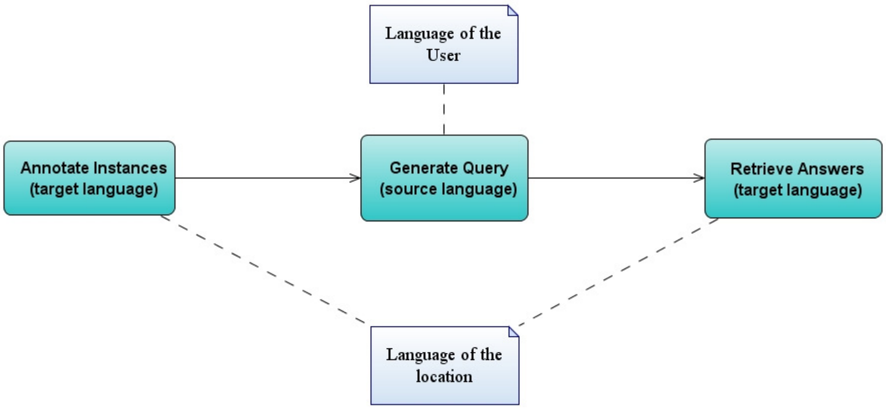
Language of the User
Annotate Instances (target language)
Generate Query (source language)
Retrieve Answers (target language)
Language of the location

Figure 2. Cross-language Setting

facilitate retrieval using SPARQL query language (Prud'hommeaux and Seaborne, 2006). Question patterns corresponding to one or several ontological relations were produced after questions for users were collected and used in the entailment module. The question patterns used by the system are very similar to those presented in the previous section and contain placeholders for the actual entities that are expected to appear in a question.

The SPARQL query associated with a pattern selected for a user question is used to retrieve the answers from the knowledge base and prepare for presentation. Given that our system is not limited to returning only textual information, further processing can be applied to the retrieved data. For example, for proximity questions the list of answers consists of cinema names and their GPS-coordinates, which are used by the *Answer Sorting* component to reorder the list of answers on the basis of their distance to the user's location.

Besides presenting the possible answers to a given question, the system can offer additional information based on the answer's type:

- a map for answers that are location names,
- a route description for answers that are cinema names,
- a video-trailer for answers that are movie names and
- an image for answers that are person names.

Due to the fact that a common semantics is shared by all four languages by way of a domain ontology, the system can be used not only in a monolingual setting, but also in a cross-language setting. This corresponds to a user-scenario where a tourist asks for information in their own language in a foreign location (i.e. English speaker in Italy). The only difference between monolingual and cross-language scenarios is that in the cross-language setting, the *QA Core* subsystem (Figure 1) selects a *Find Entailed Relation* component according to the user input's language. This is due to the entailment algorithms that tend to use language specific resources in order to attain high accuracy results of matching the user request with one of the lexicalized relations (patterns). It is only the entailment component that has to be provided in order to adapt the system to new input languages, once the lexicalized relations have been translated either manual or automatically.

Both the *Instance Annotator* and the *Answer Retriever* are language independent, but location dependent (Figure 2). The *Answer Retriever* depends on the location since it is querying data found at that place (i.e. Italy), while the *Instance Annotator* looks up instances of the data in the user's question (i.e. annotates an English question). They are language independent since they are working with data abstractions like SPARQL queries (*Answer Retriever*) or work at character level and do not consider language specific aspects, like words, in their look-up process (*Instance Annotator*).

The current version of the system[1] is designed according to the SOA (Service Oriented Architecture) and is implemented as point-to-point integrated web services. Any of the system's components can be substituted by alternative implementations with no need for further changes as long as the functionality remains the same.

[1] http://attila.dfki.uni-sb.de:8282/ QallMe_Prototype_WEB_Update/faces/Page6.jsp

4 Evaluation

A preliminary evaluation of the first prototype was carried out on randomly selected questions from a benchmark specifically designed for the project. This benchmark was developed to contain questions about various aspects from the domain of tourism and for this reason we filtered out questions not relevant to cinema or movies. The evaluation of the system did not assess whether it can extract the correct answer. Instead, it measured to what extent the system can select the right SPARQL pattern. The explanation for this can be found in the fact that once a correct question pattern is selected, the extraction of the answer requires only retrieval of the answer from the database. Moreover, it should be pointed out that the main purpose of this preliminary evaluation was to test the interaction between components and indicate potential problems, and it was less about their performances.

Table 1 summarises the results of the evaluation. The number of questions used in the evaluation is different from one language to another. This can be explained by the fact that for each language a number of questions (in general 500) was randomly selected from the benchmark and only the ones which referred to cinema or movies were selected. The column *Questions* indicates the number of questions assessed. The *Correct* column indicates for how many questions a correct SPARQL was generated. The *Wrong* column corresponds to the number of questions where a wrong or incomplete SPARQL was generated. This number also includes cases where no SPARQL was generated due to lack of corresponding answer pattern.

	Questions	Correct	Wrong
English	167	74 (44.31%)	93 (55.68%)
German	214	120 (56.04%)	94 (43.92%)
Spanish	58	50 (86.20%)	8 (13.79%)
Italian	99	46 (46.46%)	53 (53.53%)

Table 1: Evaluation results

As can be seen, the results are very different from one language to another. This can be explained by the fact that different entailment engines are used for each language. In addition, even though the benchmark was built using a common set of guidelines, the complexity of questions varies from one language to another. For this reason, for some questions it is more difficult to find the correct pattern than for others.

Analysis of the results revealed that one of the easiest ways to improve the performance of the system is to increase the number of patterns. Currently the average number of patterns per language is 42. Improvement of the entailment engines is another direction which needs to be pursued. Most of the partners involved in the project have more powerful entailment engines than those integrated in the prototype which were ranked highly in RTE competitions. Unfortunately, many of these engines cannot be used directly in our system due to their slow speed. Our system is supposed to give users results in real time which imposes some constraints on the amount of processing that can be done.

5 Conclusions

This paper presented the first prototype of an entailment-based QA system, which can answer questions about movies and cinema. The use of a domain ontology ensures that the system is cross-language and can be extended to new languages with slight adjustments at the entailment engine. The system is implemented as a set of web services and along a Service Oriented Architecture.

6 Acknowledgements

This work is supported by the EU-funded project QALL-ME (FP6 IST-033860).

References

Androutsopoulos, I. and G.D. Ritchie and P. Thanisch. 1995. *Natural Language Interfaces to Databases -- An Introduction*, Journal of Natural Language Engineering, vol.1, no.1, Cambridge University Press.

Popescu Ana-Marie, Oren Etzioni, and Henry Kautz. 2003. *Towards a theory of natural language interfaces to databases*. In Proceedings of the conference on Intelligent User Interfaces.

Dagan Ido and Oren Glickman. 2004. *Probabilistic textual entailment: Generic applied modeling of language variability.* In PASCAL Workshop on Learning Methods for Text Understanding and Mining, Grenoble.

Ou Shiyan, Viktor Pekar, Constantin Orasan, Christian Spurk, Matteo Negri. 2008. *Development and alignment of a domain-specific ontology for question answering*. In Proceedings of the 6th Edition of the Language Resources and Evaluation Conference (LREC-08).

Prud'hommeaux Eric, Andy Seaborne (eds.). 2006. *SPARQL Query Language for RDF*. RDF Data Access Working Group.