Multi-source hybrid Question Answering system

Seonyeong Park, Hyosup Shim, Sangdo Han, Byungsoo Kim, Gary Geunbae Lee

Pohang University of Science and Technology, Pohang, Republic of Korea
{sypark322, hyosupshim, hansd, bsmail90, gblee}@postech.ac.kr

Abstract. In this demonstration, we present a multi-source hybrid Question Answering (QA) system. Our system consists of four sub-systems. 1) a knowledge base based QA, 2) an information retrieval based QA, 3) a keyword QA and 4) an information-extraction to construct our own knowledge base from web texts. With these sub-systems, we can query three types of information sources: curated knowledge bases, automatically-constructed knowledge bases and wiki texts.

Keywords: knowledge base based QA, information retrieval based QA, information extraction, Keyword QA, knowledge base construction

1 Introduction

Various approaches to QA based on knowledge base (KB) have been proposed. QA is evolving from systems based on information retrieval (IR) to systems based on KBs. QA based on KBs gives very high precision, but requires curated KBs; but these KBs cannot cover all the information that web text can convey. To solve this limitation, multiple information sources other than curated KBs are needed. In this demo, we present a hybrid QA system (Fig 1) that uses multiple information sources: a curated KB, an automatically-constructed KB, and web text.

2 System description

2.1 Knowledge base based QA

A knowledge base based Question Answering (KB-based QA) system that takes a natural language (NL) question as its input and retrieves its answer from structured (possibly curated) KBs like DBpedia and Freebase. A KB-based QA system uses highly-structured information sources, so it produces very specific answer sets.

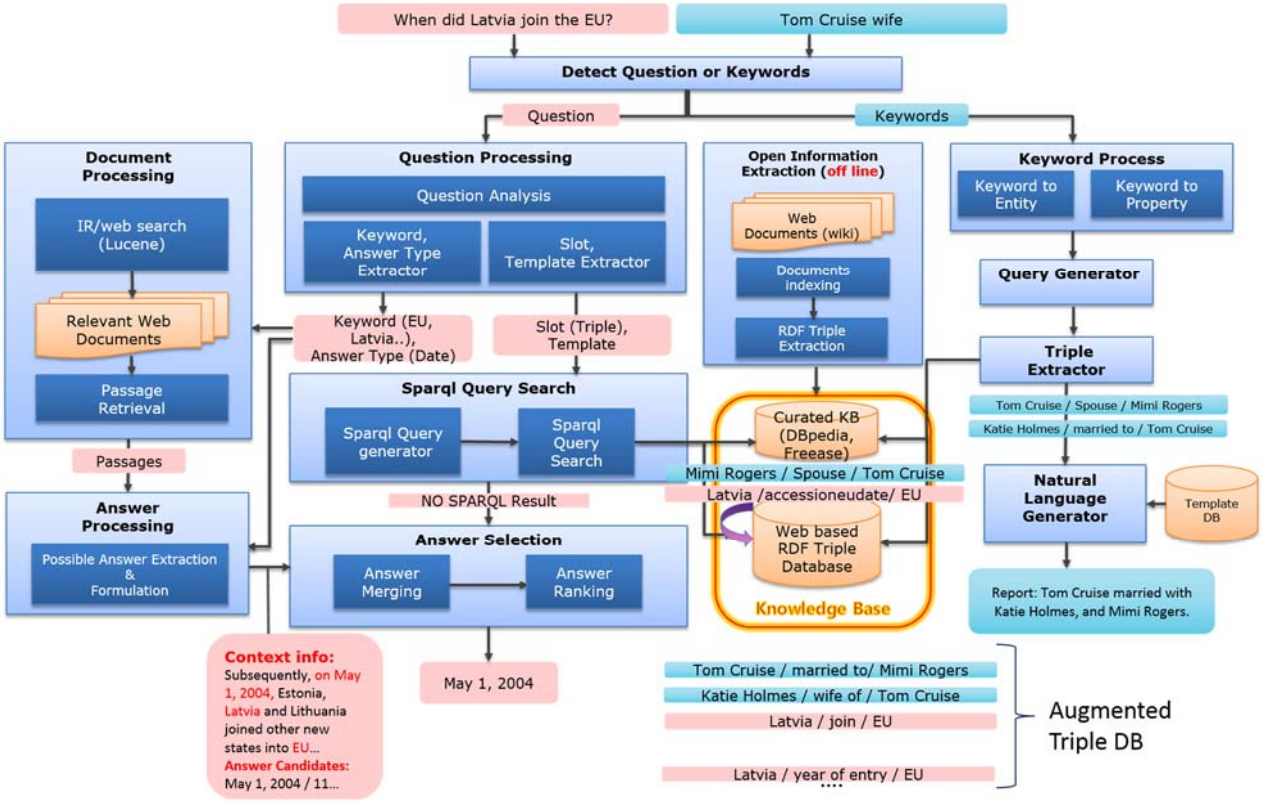

Fig 1. Architecture of entire system

We take two approaches to handle the task. The first approach uses semantic parsing [1]; the other uses Lexico-Semantic patterns (LSP) matching. In the semantic parsing approach, we first use a beam segmenter to generate candidate segmentations of a NL question; then we use string based methods and automatically generated <NL phrases, KB node mapping dictionary> to try to match KB vocabulary to the segments. We generate query candidates by using a small set of hand-crafted grammar rules to combine segments into a single formal representation of meaning. In the LSP approach, we generate patterns that consist of regular expression patterns that describe the lexical / POS / chunk type patterns of a NL question and a SPARQL query template. If a match is found, slots in the SPARQL query template are filled with the word-matched chunks from NL question. However, a KB-based QA modules has no context information, and therefore cannot rank its answer candidates; instead KB-based QA passes its answer candidate to an answer merging module in the IR-based QA and lets the module rank the answer candidates.

2.2 Information Retrieval based QA

An information retrieval based Question Answering (IR-based QA) system searches text to find answers. Our IR-based QA includes four modules (Fig 1): the first classifies answer type and analyzes the question semantically; the second retrieves passages by searching documents that are related to the user question; the third extracts answer candidates; the fourth merges answer candidates from IR-based QA and KB-based QA, scores the answer candidates and returns the final list of answers. The difference between our system and other systems is that our system uses context information to scores answer candidates which are the results of the SPARQL not only from IR-based answer candidate extraction.

We used Ephyra[1] for question processing, which includes extracting keywords by lexical, syntactic and semantic analysis; and a hybrid answer-type classifier that uses rules and a classifier based on machine learning [2]. We also used Lucene[2] for indexing wiki pages dump and for searching documents and passages related to the answers. After passages are searched, sentences in the passages are scored. We extract named entities which have the same or similar answer types as answer candidates from n-best sentences in passages. Finally, using semantic relatedness among question and sentences that include answer candidates, our system ranks answer candidates from IR-based and KB-based modules. End of that process, the system gives the final answer list to user.

[1] https://www.ephyra.info

[2] http://lucene.apache.org/core

2.3 Keyword QA

The keyword QA system is a system takes keywords query input and returns an NL report as the result. Because the query is combination of keywords, multiple triples are extracted as answers. Because being required to evaluate multiple triples can confuse or irritate users, the system generates an NL report from the extracted triples.

The system matches each keyword to an entity or property in the KB to extract answer from it. To match keywords to entities or properties, we used AIDA [3] and ESA[3] module. First, AIDA tries to match each keyword to an entity. When keywords are not matched to an entity, we use the ESA module to match them to a property. A rule-based query generator generates a query to extract data from a KB. We used manually-generated NL generation templates to generate an NL report. The NL generation templates are structured as property-predicate pairs.

We generated keyword 670 queries to evaluate the accuracy of the keyword QA module. Our system returned a correct[4] answer for 95.1% of these questions.

2.4 Knowledge base by open information extraction

Although a KB has a large data capacity, it can only cover small amount of information compared to its original free text. To handle this problem, we constructed a repository that consists of triples extracted from free text. We exploit the dependency tree and semantic role labels (SRL) of a sentence to extract triples from free text.

We defined ‘extraction templates’ that specify how triples should be extracted for each dependency tree structure pattern. To generate extraction templates automatically, we used bootstrapping methods. A whole document is retrieved to find sentences that contain word tokens that appear in arguments and relation words of each seed triple. Then we constructed a dependency tree of the sentence for each seed triple, sentence pair, and identified a linear path which contains arguments and relation words. This path with position of arguments and relation words in the path can produce an extraction template.

SRL outputs similar results that can be transformed to triple format. Predicates of the results are regarded as relation phrases and each argument and argument modifier are regarded as each argument of triples. We also used a small set of rules to transform SRL results to triples.

---

[3] http://ticcky.github.io/esalib

[4] "Correct" means the result was reasonable interpretation for the keyword query based on human judgment

2.5 Integration

Our system can process both NL question and keywords. Because NL question and keywords are processed by different sub-systems, our system has a module that disambiguates the query form. The module identifies whether a user query is NL question or a keywords. To disambiguate them, we trained a model based on a conditional random field algorithm. Our training data are our query data that include NL questions and keywords. Our features are an n-gram of words and POS tags.

Our system uses semantic relatedness among question and sentences which include answer candidates to rank answer candidates from KB-based QA and IR-based QA. At the end of the process, the system gives the final answer list to user.

Acknowledgments This work was supported by ICT R&D program of MSIP/IITP. [10044508, Development of Non-Symbolic Approach-based Human-LikeSelf-Taught Learning Intelligence Technology] and ATC (Advanced Technology Center) Program - "Development of Conversational Q&A Search Framework Based On Linked Data: Project No. 10048448".

References

[1] Berant, J., Chou, A., Frostig, R., & Liang, P. (2013, October). Semantic Parsing on Freebase from Question-Answer Pairs. In EMNLP (pp. 1533-1544)

[2] Schlaefer, N., Ko, J., Betteridge, J., Pathak, M. A., Nyberg, E., & Sautter, G. (2007). Semantic Extensions of the Ephyra QA System for TREC 2007. In TREC.

[3] Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., ... & Weikum, G. (2011, July). Robust disambiguation of named entities in text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (pp. 782-792). Association for Computational Linguistics.