Question Answering System using Multiple Information Source and Open Type Answer Merge

Seonyeong Park, Soonchoul Kwon, Byungsoo Kim, Sangdo Han, Hyosup Shim, Gary Geunbae Lee
Pohang University of Science and Technology, Pohang, Republic of Korea
{sypark322, theincluder, bsmail90, hansd, hyosupshim, gblee} @postech.ac.kr

Abstract

This paper presents a multi-strategy and multi-source question answering (QA) system that can use multiple strategies to both answer natural language (NL) questions and respond to keywords. We use multiple information sources including curated knowledge base, raw text, auto-generated triples, and NL processing results. We develop open semantic answer type detector for answer merging and improve previous developed single QA modules such as knowledge base based QA, information retrieval based QA.

1 Introduction

Several massive knowledge bases such as DBpedia (Auer et al., 2007) and Freebase (Bollacker et al., 2008) have been released. To utilize these resources, various approaches to question answering (QA) on linked data have been proposed (He et al., 2014; Berant et al., 2013). QA on linked data or on a knowledge base (KB) can give very high precision, but because KBs consist of fragmentary knowledge with no contextual information and is powered by community effort, they cannot cover all information needs of users. Furthermore, QA systems achieve low precision when disambiguating question sentences in to KB concepts; this flaw reduces QAs' performance (Yih et al., 2014).

A QA system can understand a natural language (NL) question and return the answer. In some ways, perfection of QA systems is the final goal of information retrieval (IR). Early QA systems were IR-based QAs (IRQAs). However, as large KBs such as DBpedia and Freebase have been constructed, KB-based QA (KBQA) has become increasingly important (Lehmann et al., 2015; Unger et al., 2012).

These two kinds of QA systems use heterogeneous data; IRQA systems search raw text, whereas KBQA systems search KB. KBQA systems give accurate answers because they search from KBs curated by humans. However, they cannot utilize any contextual information of the answers. The answers of IRQA are relatively less accurate than those of KBQA, but IRQA systems utilize the contextual information of the answers.

We assert that a successful QA system will require appropriate cooperation between a KBQA and an IRQA. We propose a method to merge the KBQA and the IRQA systems and to exploit the information in KB ontology-based open semantic answer type to merge the answers from the two systems, unlike previous systems that use a predetermined answer type. We improve our previous system (Park et al., 2015).

Also we can answer not only complete NL sentence questions, and questions composed of only keywords, which are frequently asked in real life. We suggest strategies and methods (Figure 1) to integrate KBQA, IRQA, and keyword QA.

2 System Architecture

2.1 KB-based QA

A KBQA system takes an NL question sentence as the input and retrieves its answers from the KBs. Because the KBs (i.e., the information sources), are highly structured, the KBQA system can produce very pin-pointed answer sets.

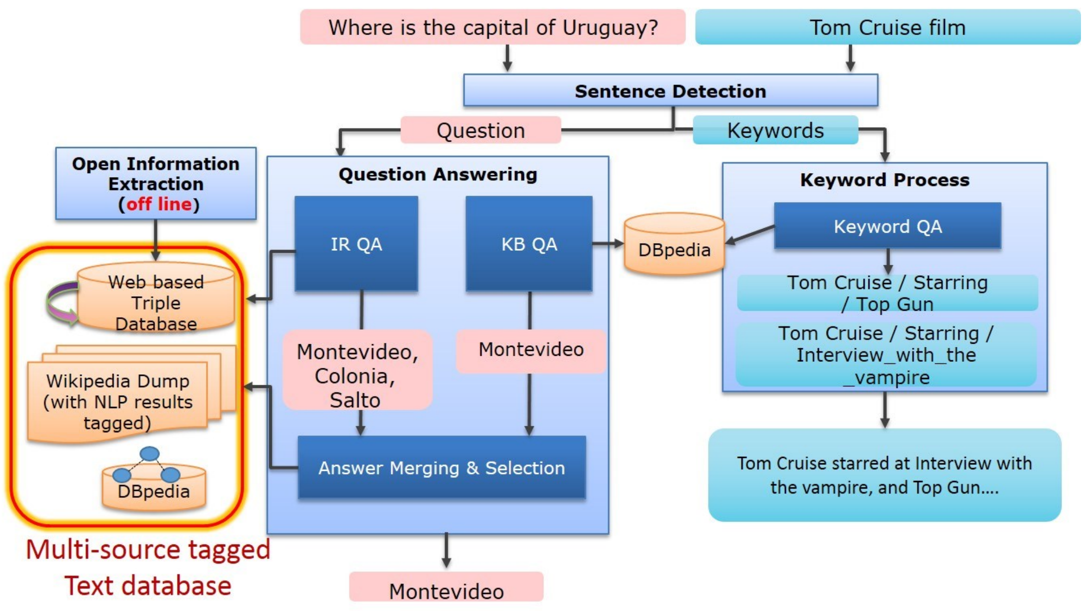
Where is the capital of Uruguay?
Tom Cruise film
Sentence Detection
Question
Keywords
Open Information Extraction (off line)
Question Answering
IR QA
KB QA
Keyword Process
Keyword QA
DBpedia
Web based Triple Database
Wikipedia Dump (with NLP results tagged)
DBpedia
Montevideo, Colonia, Salto
Montevideo
Tom Cruise / Starring / Top Gun
Tom Cruise / Starring / Interview_with_the _vampire
Answer Merging & Selection
Tom Cruise starred at Interview with the vampire, and Top Gun....
Multi-source tagged Text database
Montevideo

Figure 1. Proposed System Architecture

We combined two approaches to make this system possible. The first approach is based on semantic parsing (Berant et al., 2013), and the second is based on lexico-semantic pattern matching (Shim et al., 2014).

In the semantic parsing approach, the system first generates candidate segments of the question sentence and tries to match KB vocabularies to the segments by combining use of string-similarity based methods and an automatically generated dictionary that consists of pairs of NL phrase and KB predicate (Berant et al., 2013). Finally the system generates query candidates by applying the segments to a small set of hand-crafted grammar rules to generate a single formal meaning representation (Berant et al., 2013).

In the lexico-semantic pattern approach, we use simple patterns paired with a formal query template. The patterns consist of regular expression pattern that describes lexical, part-of-speech (PoS), and chunk-type patterns of a question sentence (Shim et al., 2014). Then the templates paired with these patterns are equipped with methods to extract information from the sentence and to fill the information into the template.

KBQA can assess the answers even when it has little or no additional contextual information, whereas other systems like IRQA systems can rely on the context from which it is retrieved (Schlaefer et al., 2007). Instead, type information and its hierarchy defined in the KB are good sources of contextual information that the KBQA can exploit. However, not all the entities defined in the KB have specific type information; therefore, relying only on the type information can reduce precision (Krishnamurthy and Mitchell, 2014).

When KBQA systems fail, it is usually due to incorrect disambiguation of entities, or to incorrect disambiguation of predicate. Both types of failures result in production of answers of the wrong types. For example, for a question sentence "*What sport does the Toronto Maple Leafs play?*" evoke answers about the arena in which the team plays, instead of the sport that the team plays, when the KBQA system fails in disambiguation.

2.2 IR-based QA

The system uses a multi-source tagged text database which is a combination of raw text, auto-generated triples, co-reference results, named entity disambiguation results, the types of named entities, and syntactic and semantic NLP results including semantic role label, dependency parser results, PoS tag. The system uses clearNLP[1] for syntactic and semantic NLP, Stanford Co-reference tool[2] for co-reference tagging, Spotlight (Mendes et al., 2011) for disambiguated named entity tagging, and SPARQL queries (e.g. *"SELECT*

[1] http://clearnlp.wikispaces.com/
[2] http://nlp.stanford.edu/

DISTINCT ?uri WHERE { res:Nicole_Kidman rdf:type ?uri. }") for tagging DBpedia ontology class types that correspond to entities, and triples that correspond to the sentence. As a result, from a sentence "*Kim was born in 1990 in Bucheon, Gyeonggi, and moved to Gunpo when she was six years old*", the system tags several triples such as < Kim; was born in; 1990 >, < Kim; was born in; Bucheon >, < Kim; was born in; Gyeonggi >, and < Kim; moved to; Gunpo >.

Our IRQA system consists of five parts similar to the architecture of our previous system (Park et al., 2015): the first part detects the semantic answer type of the question and analyzes the question; the second part generates the queries; the third part retrieves passages related to the user question; the fourth part extracts answer candidates using type checking and semantic similarity; and the last part ranks the answer candidates. The system analyzes questions from diverse aspects: PoS tagger, dependency parser, semantic role labeler, our proposed open Information extractor, and our semantic answer type detector. The system expands query using resources such as Wordnet[3] and dictionary.

The system uses Lucene[4] to generate an index and search from multi-source tagged text database. This is an efficient method to search triples and their corresponding sentences, instead of searching the raw text. Using Lucene, the system searches raw sentences and the auto-generated triples at the same time, but may find different sentences due to information loss during extraction of triples. These sentences are scored by measuring semantic similarity to the user query. From these sentences, the system extracts the named entities and compares the semantic answer type of the question to the types of these named entities (Figure 2.). Alongside the answer type, the system uses contextual information of the corresponding sentences of the answer candidates. By combining these two methods, the system selects answer candidates.

2.3 Keyword QA

Keyword QA takes a keyword sequence as the input and returns a NL report as the answer. The system extracts answer triples from the KB from the user input keyword sequences. The system uses previously generated NL templates to generate an NL report (Han et al., 2015).

2.4 Open Information Extraction

Despite their enormous data capacity, KBs have limitation in the amount of knowledge compared to the information on the web. To remedy this deficiency, we construct a repository of triples extracted from the web text. We apply the technique to the English Wikipedia[5] for the demo, but the technique is scalable to a web corpus such as ClueWeb[6]. Each triple is composed of the form < *argument1*; *relation*; *argument2* >, where the arguments are entities in the input sentence and the relation represents the relationship between the arguments.

The system integrates both dependency parse tree pattern and semantic role labeler (SRL) results of each input sentence when extracting the triples. The dependency parse tree patterns are used to generalize NL sentences to abstract sentence structures because the system can find unimportant word tokens can be ignored in the input sentence. We define how triples should be extracted for each dependency pattern. If a certain dependency pattern is satisfied, the word tokens in the pattern constitute the head word of each relation and argument s in the triple. We call these patterns 'extraction templates'. Since manual construction of extraction templates costs too much, we automatically construct them by bootstrapping with seed triples extracted from simple PoS tag patterns.

For each sentence, the SRL annotates the predicate and the arguments of the predicate with their specific roles in the sentence. The predicate is regarded as *relation* and the arguments are regarded as *argument1* and *argument2*, according to their roles. We manually define conversion rules for each SRL result.

3 Methods for Integration

3.1 Detecting Keywords and Sentence

Our system disambiguates whether the user input query is a sentence or a keyword sequence. To disambiguate a sentence, the system uses bi-gram PoS

[3] http://sourceforge.net/projects/jwordnet/
[4] http://lucene.apache.org/core/
[5] http://dumps.wikimedia.org/backup-index.html
[6] http://www.lemurproject.org/clueweb12.php/

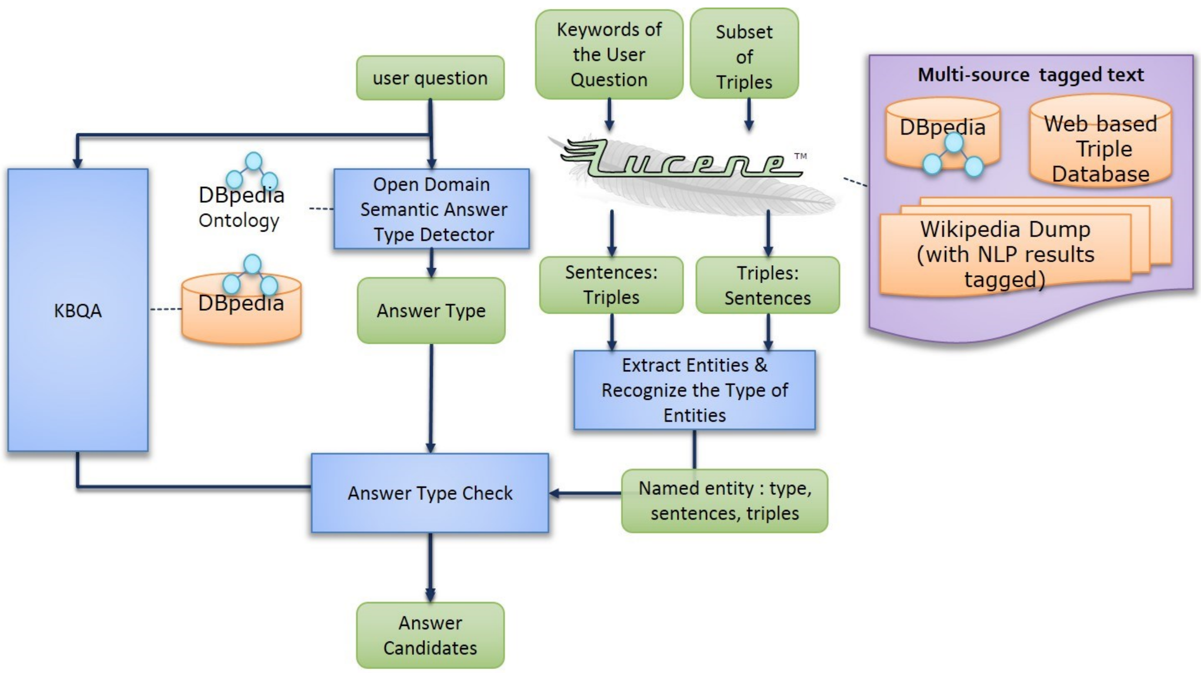
user question
Keywords of the User Question
Subset of Triples
Multi-source tagged text
DBpedia
Web based Triple Database
Wikipedia Dump (with NLP results tagged)
Lucene
DBpedia Ontology
Open Domain Semantic Answer Type Detector
KBQA
DBpedia
Answer Type
Sentences: Triples
Triples: Sentences
Extract Entities & Recognize the Type of Entities
Answer Type Check
Named entity : type, sentences, triples
Answer Candidates

Figure 2. Semantic answer type detector used to merging answer candidates

tag features and a maximum entropy algorithm. Our dataset includes 670 keyword sequences and 4521 sentences. Based on five-fold cross validation, our system correctly detected 96.27 % of the keyword sequences and 98.12 % of the sentences.

When the user query is a sentence, the query is sent to the KBQA/IRQA system. Otherwise the query is sent to the keyword QA system.

3.2 Open Semantic Answer Type detector

The proposed system integrates the answers from the KB and the multi-source tagged text database including the auto-generated triple database. Therefore the KBQA and the IRQA must share an answer-type taxonomy. A previous answer type classification task used UIUC answer type including six coarse-grained answer types and 50 fine-grained answer types (Li et al., 2002). Instead, we use the DBpedia class type hierarchy as the open semantic answer type set. The proposed semantic answer type detector involves three steps.

1. Feature Extraction: The proposed system uses the dependency parser and PoS tagger to extract the main verb and the focus. If the question is "*Who invented Macintosh computer?,*" the main verb is *'invented'* and the focus is *'who'*. The answer sentence is constructed by replacing the focus with the answer candidate and changing to declarative sentence with period, when the focus is substituted with the answer. The system can detect also whether the focus is the subject or the object of the main verb.
2. Mapping property: The system measures the semantic similarity between '*invented*' and DBpedia properties. The system determines that the most similar DBpedia property to *'invented'* is 'patent'.
3. Finding semantic answer type: The system can get the type of the subject and the object of the DBpedia property 'patent'. If the focus is the object of the property, the semantic answer type is the type of the object of the property; otherwise it is the type of the subject of the property.

If the system cannot find the answer type by these steps, the system uses an answer type classifier as in Ephyra (Schlaefer et al, 2007) and uses a transformation table to map their answer type classes in the UIUC answer type (Li et al., 2002) taxonomy to DBpedia class ontology.

3.3 Answer Merging and Re-ranking

This integrated system gets the answer candidates from both the KBQA and the IRQA. The system extracts *n*-best sentences including the answer candidates from the KBQA and the keywords from the user query.

The DBpedia types of the answer candidates from both the KBQA and the IRQA can be detected and compared to the semantic answer type (Figure 2.).

Finally, the system selects the final answer list by checking the answer types of the user query and the semantic relatedness among the answer sentence substituted focus with the answer candidates, and the retrieved sentences.

4 Conclusion

We have presented a QA system that uses multiple strategies and multiple sources. The system can answer both complete sentences and sequences of keywords. To find answers, we used both a KB and multi-source tagged text data. This is our baseline system; we are currently using textual entailment technology to improve merging accuracy.

Acknowledgments

This work was supported by the ICT R&D program of MSIP/IITP [R0101-15-0176, Development of Core Technology for Human-like Self-taught Learning based on a Symbolic Approach].

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. *Dbpedia: A nucleus for a web of open data*. Proceedings of the Sixth international The semantic web and Second Asian conference on Asian semantic web conference (pp. 722-735).

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. *Semantic Parsing on Freebase from Question-Answer Pairs*. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. 1533-1544.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. *Freebase: a collaboratively created graph database for structuring human knowledge*. Proceedings of the 2008 SIGMOD international conference on Management of data. 1247-1250.

Sangdo Han, Hyosup Shim, Byungsoo Kim, Seonyeong Park, Seonghan Ryu, and Gary Geunbae Lee. 2015. *Keyword Question Answering System with Report Generation for Linked Data*. Proceedings of the Second International Conference on Big Data and Smart Computing.

Shizhu He, Kang Liu, Yuanzhe Zhang, Liheng Xu, and Jun Zhao. 2014. *Question Answering over Linked Data Using First-order Logic*. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. 1092-1103.

Jayant Krishnamurthy and Tom M. Mitchell. 2014. *Joint Syntactic and Semantic Parsing with Combinatory Categorial Grammar*. Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics. 1188-1198.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. *DBpedia – A large-scale, multilingual knowledge base extracted from Wikipedia*. Semantic Web: 6(2). 167-195.

Xin Li, Dan Roth, *Learning question classifiers*. 2002. Proceedings of the 19th international conference on Computational linguistics-Volume 1. 1-7.

Pablo N. Mendes, Max Jakob, Andrés García-Silva , and Christian Bizer. 2011. *DBpedia Spotlight: Shedding Light on the Web of Documents*. Proceedings of the 7th International Conference on Semantic Systems. 1-8.

Seonyeong Park, Hyosup Shim, Sangdo Han, Byungsoo Kim, and Gary Geunbae Lee. 2015. *Multi-source hybrid Question Answering system*. Proceeding of The Sixth International Workshop on Spoken Dialog System

Nico Schlaefer, Jeongwoo Ko, Justin Betteridge, Guido Sautter, Manas Pathak, and Eric Nyberg. 2007. *Semantic Extensions of the Ephyra QA System for TREC 2007*. Proceedings of the Sixteenth Text REtrieval Conference.

Hyosup Shim, Seonyeong Park, and Gary Geunbae Lee. 2014. *Assisting semantic parsing-based QA system with lexico-semantic pattern query template*. Proceedings of Human and Cognitive Language Technology. 255-258.

Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. 2012. *Template-based question answering over RDF data*. Proceedings of the 21st international conference on World Wide Web. 639-648.

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. *Semantic parsing for single-relation question answering*. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics. 643-648.