KbQAS: A Knowledge-based QA System

Dat Quoc Nguyen, Dai Quoc Nguyen, and Son Bao Pham

Faculty of Information Technology
University of Engineering and Technology
Vietnam National University, Hanoi
`{datnq, dainq, sonpb}@vnu.edu.vn`

Abstract We present the first ontology-based Vietnamese QA system KbQAS where a new knowledge acquisition approach for analyzing English and Vietnamese questions is integrated.

1 Introduction

Recent years have witnessed a new trend of building ontology-based question answering (QA) systems to make the use of semantic information in terms of semantic web. This demo paper introduces a knowledge-based QA system named KbQAS, the *first* ontology-based QA system for Vietnamese. The target domain is modeled as an ontology in our KbQAS system to leverage techniques and latest advances in the semantic web. Thus semantic markups can be used to add meta-information to provide more precise answers to complex questions expressed in natural language. This is an avenue that has not been actively explored for Vietnamese.

2 System overview

The architecture of our KbQAS system as shown in figure 1 contains a question analysis component where *a new knowledge acquisition approach* to systematically build knowledge bases of grammar rules for processing natural language questions over semantic annotations is integrated. The question analysis component takes the user question as an input and returns an intermediate element representing the question in a compact form. The answer retrieval component is responsible for making sense of the user query with respect to a target ontology using concept-matching techniques for a natural language phrase and elements in the ontology. It takes an intermediate representation produced by the question analysis component and an ontology as its input to generate answers.

Illustrative example. For demonstration[1] purpose, we exploit an ontology modeling the organizational structure of an university as referred in our first KbQAS version [1].

Consider the complex structure question of *"Liệt kê tất cả sinh viên học lớp K50 khoa học máy tính mà có quê ở Hà Nội?"* ("List all students studying in K50 computer science course, who have hometown in Hanoi?"): the Question analysis component determines that this question has a query structure of type *"And"* and two query-tuples (Normal, List, sinh viên$_{student}$, học$_{study}$, lớp K50 khoa học máy tính$_{K50\ computer\ science\ course}$, $?$[2]) and (Normal, List, sinh viên$_{student}$, có quê$_{has\ hometown}$, Hà Nội$_{Hanoi}$, $?$).

Query-tuples are then mapped to ontology-tuples by the Ontology mapping module in the Answer retrieval component: (sinh_viên$_{student}$, học$_{study}$, lớp_K50_khoa_học_

---

[1] KbQAS available in http://cntt.dyndns.info:8856/KbQAS/

Vietnamese question analysis demonstration at http://cntt.dyndns.info:8856/KbVnQA/

English question analysis demonstration available in http://cntt.dyndns.info:8856/KbEnQA/

[2] $?$ indicates the missing element.

máy_tính$_{K50\_computer\_science\_course}$) and (sinh_viên$_{student}$, có_quê$_{has\_hometown}$, Hà_ Nội$_{Hanoi}$). With each ontology-tuple, the Answer extraction module finds all satisfied instances in the target ontology before generating an answer presented in the figure 1 based on the question structure *"And"* and the question class *"List"*.

Figure 1: The system's architecture and illustrations of question analysis and question answering.

Intermediate representation of question. The intermediate representation used in our KbQAS system consists of a *question-structure* and one or more *query-tuple*s in the following format: *(sub-structure, question-class, $Term_1$, Relation, $Term_2$, $Term_3$)*. Simple questions only have one query-tuple and its question-structure is the query-tuple's sub-structure. More complex questions such as composite questions have several sub-questions, each sub-question is represented by a separate query-tuple, and the question-structure is to capture this composition attribute.

Question analysis component. The question analysis component contains three modules: preprocessing, syntactic analysis and semantic analysis. It makes the use of JAPE grammars in GATE framework [2] to specify regular expression patterns based on semantic annotations for question analysis. The preprocessing and syntactic modules are responsible for identifying noun phrases, question-phrases, and the relations among noun phrases or between noun phrase and question-phrase in the input questions. The semantic analysis module embodies **the key innovation** in the current KbQAS version. This semantic module utilizes the noun phrase, question-phrase and relation annotations created by the two preceding modules. It aims to specify the question-structure and to produce the query-tuples as the intermediate representation of the input question.

In the current semantic analysis module, we propose *a new knowledge acquisition approach* for analyzing natural language questions by applying Single Classification Ripple Down Rules (SCRDR) methodology [3] to acquire rules incrementally. A SCRDR knowledge base, where grammar rules over semantic annotations are structured in an exception structure and new rules are only added to correct errors of existing rules, is built to generate the intermediate representations of questions. This process is to create rules in a systematic manner to solve difficulties which appear in such most existing rule-based question analysis approaches as in Aqualog system [4] and the first KbQAS version [1] in managing the interaction between rules and keeping consistency among them. Moreover, our proposed approach enables ones to easily construct a new

system or adapt an existing system to a new domain or a new language, thus a lot of time and effort of human experts can be saved. The experimental evaluation of our method for English and Vietnamese question analyses is detailed in our previous work [5].

Answer retrieval component. The detail description of this component can be found in the first KbQAS version [1]. In short, the task of its Ontology mapping module is to map terms and relations in the query-tuples to concepts, instances and relations in the target ontology. Then the Answer extraction module finds all instances associated to mapped instances and concepts, satisfying ontology relations. Depending on the question-structure and question-class, the best semantic answer will be returned.

Evaluation. The performance of the current KbQAS on a wide range of different Vietnamese questions is promising with accuracies of 84.1% and 82.4% accounted for the question analysis and answer retrieval components, respectively.

3 Demonstration: knowledge acquisition for question analysis

In this section, we only illustrate the process of systematically constructing a SCRDR knowledge base for analyzing English questions[3]. In demonstration session, however, we plan to present other illustrations of building English and Vietnamese knowledge bases for question analysis, and to provide other illustrative examples of the KbQAS.

The following exemplification shows how the knowledge base building process works. When we encounter the question *"who are the researchers in semantic web research area ?"* ([QuestionPhrase: who] [Relation: are the researchers in] [NounPhrase: semantic web research area]). Supposed we start with an empty knowledge base, the fired rule (i.e. last satisfied rule) is the default rule[4] that gives empty conclusion. This can be corrected by adding the following exception rule to the knowledge base:

Rule: R1
(({QuestionPhrase}):QPhrase ({Relation}):Rel ({NounPhrase}):NPhrase):left
--→ :left.RDR1_ = {category1 = "UnknTerm"}
, :QPhrase.RDR1_QP = {} , :Rel.RDR1_Rel = {} , :NPhrase.RDR1_NP = {}
Conclusion: question-structure *"UnknTerm"* and query-tuple (RDR1_.category1, RDR1_QP.QuestionPhrase.category, ?, RDR1_Rel, RDR1_NP, ?).

If the condition of rule R1 matches whole input question, a new annotation RDR1_ will be created to entirely cover the input question and new annotations RDR1_QP, RDR1_Rel and RDR1_NP will also be generated for covering sub-phrases of the input question. Once rule **R1** is fired, the matched input question is deemed to have a query-tuple with *sub-structure* taking the value of the feature *"category1"* of RDR1_ annotation, *question-class* taking the value of the feature *"category"* of QuestionPhrase annotation surrounding the same span as RDR1_QP annotation. Besides, the query-tuple's *Relation* is the string covered by RDR1_Rel, $Term_2$ is the string surrounded by RDR1_NP while $Term_1$ and $Term_3$ are missing.

---

[3] We utilized JAPE grammars employed in AquaLog [4] for detecting the noun-phrase, question-phrase, and relation annotations in English questions. We also reused question-class definitions and took question examples of Aqualog for building the SCRDR knowledge base.

[4] A rule is composed of a condition part and a conclusion part. A condition is a regular expression pattern over semantic annotations using JAPE grammars. The conclusion contains the question structure and the tuples corresponding to the intermediate representation where each element in the tuple is specified by a newly posted annotations from matching the rule's condition. The default rule typically contains a trivial condition which is always satisfied.

Using rule R1, the knowledge base returns a correct intermediate representation of question-structure *"UnknTerm"* and query-tuple (UnknTerm, QU-who-what, ?, researchers, semantic web research area, ?) for the encountered question.

When it comes to the question *"How many researchers work on AKT project?"* ([RDR1_: [RDR1_QP: how many researchers] [RDR1_Rel: work on] [RDR1_NP: AKT project]]), rule R1 is the fired rule. However, rule R1 gives a wrong conclusion of question-structure *"UnknTerm"* and query-tuple (UnknTerm, QU-howmany, ?, work, AKT project, ?). We can add the following exception rule R4 to correct the conclusion of rule R1 by using constrains via an additional condition:

Rule: R4

({RDR1_}):left --→ :left.RDR4_ = {category1 = "Normal"}

Condition: RDR1_QP.hasAnno == QuestionPhrase.kind == ListWhichHMany

Conclusion: question-structure *"Normal"* and query-tuple (RDR4_.category1, RDR1_QP.QuestionPhrase.category, RDR1_QP, RDR1_Rel, RDR1_NP, ?).

The additional condition of rule R4 matches a RDR1_QP annotation which has a QuestionPhrase annotation covering their substring with *"ListWhichHMany"* as the value of its feature *kind*. The extra annotation constraint *hasAnno* requires that the text covered by the annotation (e.g. RDR1_QP) must contain the specified annotation (e.g. QuestionPhrase). Rule R4 generates the correct output consisting of question-structure *"Normal"* and query-tuple (Normal, QU-howmany, researchers, work, AKT project, ?).

Turning to the question *"which projects are about ontologies and the semantic web?"* ([RDR4_: [RDR1_QP: which projects] [RDR1_Rel: are about] [RDR1_NP: ontologies]] [And: and] [NounPhrase: the semantic web]), it is satisfied by rule R4, nevertheless rule R4 results to an incorrect intermediate representation as RDR4_ annotation only covers a part of the question. The following exception rule R37 is added to rectify the conclusion of the rule R4:

Rule: R37

({RDR4_}{And}({NounPhrase}):NPhrase):left

--→ :left.RDR37_ = {category1 = "UnknRel", category2 = "UnknRel"}

, :NPhrase.RDR37_NP = {}

Condition: RDR1_Rel.hasAnno == Relation.category == Rel-Auxiliary

Conclusion: question-structure *"And"* and two query-tuples (RDR37_.category1, RDR1_QP.QuestionPhrase.category, RDR1_QP, ?, RDR1_NP, ?) and (RDR37_.category2, RDR1_QP.QuestionPhrase.category, RDR1_QP, ?, RDR37_NP, ?).

Rule R37 enables to return a correct intermediate representation for the question with question-structure *"And"* and query-tuples (UnknRel, QU-whichClass, projects, ?, ontologies, ?) and (UnknRel, QU-whichClass, projects, ?, semantic web, ?).

References

1. Nguyen, D.Q., Nguyen, D.Q., Pham, S.B.: A Vietnamese Question Answering System. In: Proc. of KSE'09, IEEE CS (2009) 26–32
2. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: Proc. of ACL'02
3. Richards, D.: Two decades of ripple down rules research. Knowledge Engineering Review **24**(2) (2009) 159–184
4. Lopez, V., Uren, V., Motta, E., Pasin, M.: AquaLog: An ontology-driven question answering system for organizational semantic intranets. Web Semantics **5**(2) (2007) 72–105
5. Nguyen, D.Q., Nguyen, D.Q., Pham, S.B.: Systematic Knowledge Acquisition for Question Analysis. In: Proc. of RANLP 2011. (September 2011) 406–412