QA SYSTEM METIS BASED ON WEB SEARCHING AND SEMANTIC GRAPH MATCHING

Dongli HAN*, Yuhei KATO**, Kazuaki TAKEHARA**, Tetsuya YAMAMOTO**, Kazunori SUGIMURA**, and Minoru HARADA**
**Department of Computer Science and System Analysis, NIHON University*
***Department of Integrated Information Technology, Aoyama Gakuin University*

Abstract: We have developed a question-answering system Metis with natural-language interface. Metis generates the answer to a question by comparing the semantic graph of the question sentence with sentences discovered on the Internet as knowledge source. Specifically, we first get a set of semantic frames for the question sentence, as the output from a semantic analysis system, SAGE. Then we extract several keywords from all semantic frames using SVM. After that we search the Web to find knowledge sentences based on the keywords and input each knowledge sentence into SAGE in order to get its semantic graphs similarly. Finally, the similarities between the semantic graph of the question sentence and that of each knowledge sentence are calculated to determine the most reliable knowledge sentence, in which a constituent is chosen as the answer to the question. An experiment to examine the effectiveness of our method showed that 65% of the questions for which suitable knowledge sentences had been found were replied correctly.

Key words: Natural Language Processing, Question Answering System, Semantic Analysis, EDR-Dictionary

1. INTRODUCTION

QA systems with natural-language-interface have been popular recently. Here are some study cases. Endo etc. made efforts in trying to handle wider question type by employing classified type of named entity [1]. Kurata etc.

adopted a measure of distance between nodes in the graph structure of the knowledge sentence to determine the answer among all candidates [2]. Sasaki etc. regarded the two components of a QA system, question analysis and answer extraction, as 2-class classification problems, then used SVM to determine the question type of a given question, and to select answer candidates that match the question type [3]. In another study, Murata etc. parsed syntactically both the question sentence and the knowledge one extracted from a database, and then extract the answer from the latter by comparing and matching their dependency structures [4].

As shown above, all studies are common in the point attempting to find the answer to the question by surface information only, without considering any semantic factors which might be quite important in this process. We believe this is the principal cause for the high ratio of wrong answers in most systems. In this study, we try to pick out a clause, or simply a word, from the knowledge sentence as the answer to the question by comparing the semantic graphs of the question sentence and the knowledge sentence. We believe the delicate comparison at the semantic level could reduce the mistakes occurred in the process.

Specifically, we first get a set of semantic frames for the question sentence, as the output from a semantic analysis system, SAGE. Then we extract several keywords from all semantic frames using a Support Vector Machine. After that we search the Web to find knowledge sentences based on the keywords and input each knowledge sentence into SAGE in order to get its semantic graphs similarly. Finally, the similarities between the semantic graph of the question sentence and that of each knowledge sentence are calculated and compared to determine the most reliable knowledge sentence, from which a constituent is chosen as the answer to the question.

2. SEMANTIC ANALYSIS

The main difference between our method and the previous ones is we attempt to find the answer to a question based on not the raw text, but the tagged ones, i.e. the semantically analyzed texts. SAGE, the semantic analysis system built by Maezawa etc. does this for us [5][6][7]. It determines the word meanings and the semantic relations among words according to the definition and statistical information registered in the EDR Dictionary[1]. Each case-frame in the analytical results corresponds to a clause in the sentence. And as shown in Figure 1, a case-frame contains tow parts:

[1] http://www.iijnet.or.jp/edr/index.html

an “f” for the clause itself, and several “s” for morphemes included in the clause.

```
f: 1,今日は,は,ME,2,,[],[],[],,[]
   s: 2,今日,キョウ,,3c5a5b,JSM,JN4,,
   s: 3,は,ハ,,,FJJ,JJO,,
f: 4,休日にも関わらず、,@renyou,DO,8,,[],[],[],,[否定]
   s: 5,休日,キュウジツ,,,FTM,JN1,,
   s: 6,に,ニ,,,KKJ,JJO,,
   s: 7,も,モ,,,FJJ,JJO,,
   s: 8,関わら,カカワラ,関わる,3c338d,DOS,JVE,子音動詞ラ行,未然形
   s: 9,ず,ズ,ぬ,,JOD,JJD,助動詞ぬ型,基本連用形
   s: 10,、,,,,TOT,JSY,,
f: 11,彼が,が,ME,12,,[],[],[ag4],,[]
   s: 12,彼,カレ,,2dc304,FTM,JN1,,
   s: 13,が,ガ,,,KKJ,JJO,,
f: 14,出勤している。,,DO,15,16,[ti1,cd4,ag11],[],[],,[非完結相]
   s: 15,出勤,シュッキン,,0f58cd,SAM,JSA,,
   s: 16,して,シテ,する,3d06c7,DOS,JVE,サ変動詞,タ系連用テ形
   s: 17,いる,イル,,,DOB,JJD,母音動詞,基本形
   s: 18,。,,,,KUT,JSY,,
e: 19,null,null,[mn14]
```

Figure 1. Case-frames for the sentence
「今日は休日にも関わらず、彼が出勤している。」
(He is working although it is not working day.)

Figure 2 is the graph generated based on the analytical results in Figure 1, and what we want for this study. Here, we expand Sowa’s concept structure [8] to produce our semantic graphs.

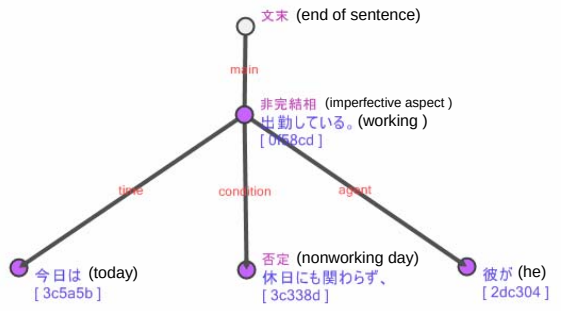


Figure 2. Semantic graph for the sentence
「今日は休日にも関わらず、彼が出勤している。」
(He is working although it is not working day.)

3. PREPARATION FOR GRAPH MATCHING

We take the following steps to prepare for matching graphs between the question sentence and the knowledge sentence.

3.1 Question type identification

Taking the case-frames of a question sentence as the input, the first thing we have to do is to identify its question type. Here in this study, we classify questions into five categories: “what”, “who”, “where”, “when”, and “other”. We determine the question type according to the EDR concept ID of the interrogative clause as shown in Table 1.

Table 1. Question types

question type	concept ID of the interrogative clause	category concept
what	0e451a, 3cf234	3aa966(concept)
who	3cfe2c, 101bab4	30f6b0(human)
where	101260 0e44fb	30f751(location), 30f746(organization) 444d86(thing)
when	0e4d47	30f776(time), 30f7e4(event)
other	1014db, 101438, 101f65, 1012f8, 101439, etc.	3aa966(concept)

For instance, if the concept ID of the interrogative clause appears as “101bab” in its case-frame, we say the question is a “who” question, and assign a category concept “human” to the interrogative clause for the sake of similarity calculation between nodes in graphs later.

3.2 Keyword extraction by SVM

We extract keywords for later web searching using SVM[2]. SVM (Support Vector Machine) is an effective machine learning mechanism, and usually used to classify some data points into two classes. Here, we follow two steps to establish our SVM models. First we select 255 question sentences containing 1657 clauses from internet or books on quiz program as the training data, and divide all clauses into two classes: valuable keywords, and non-valuable keywords by handcraft. Then, we settle four characteristic measures for the calculation of SVM: POS, deep case, distance between the interrogative clause and the keyword itself, and the length of the keyword.

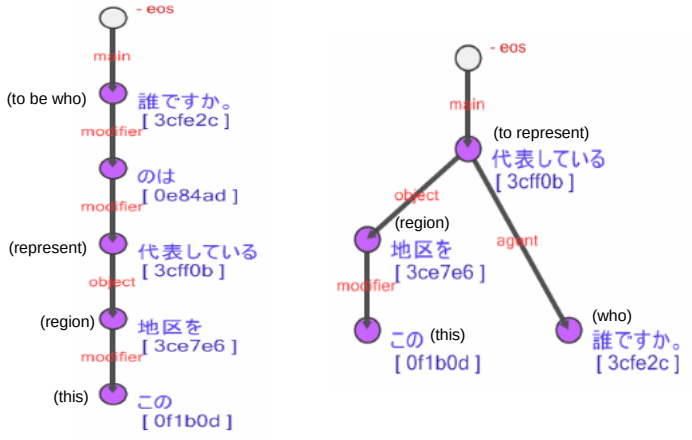
[2] http://www.chasen.org/~taku/software/TinySVM/

3.3 Web searching for knowledge

Using the extracted keywords, we conduct web searching for knowledge sentences. We use Google[3] as the search engine, and restrict the number of keywords within 4 empirically. In case the number of extracted keywords exceeds 4, combinations of every 4 keywords are tried for web searching. If no page hits with all combinations of 4-keyword, we reduce one keyword a time till hitting page appears or the number of used keywords becomes 0. Along this line, sentences in the hitting pages containing all keywords used for web searching are extracted as knowledge sentence candidates.

3.4 Paraphrase of the question sentence

According to our algorithm, we take the main predicate as the root of the graph and start graph matching from it. Thus graph matching intends to become harder if the interrogative word appears as the main predicate in the question sentence.



(a) (b)

Figure 3. Graphs before and after the paraphrase

For instance, (a) in Figure 3 shows the semantic graph of “この地区を代表しているのは誰ですか” (who is the person representing this region) where the main predicate is the interrogative word. While in most cases, a knowledge sentence will probably appear as something like “鈴木一郎がこの地区を代表している” (Suzuki Ichiro represents this region) as shown in Figure 4. Obviously, a paraphrase as shown in Figure 3 from (a) to (b) will

[3] http://www.google.com/

enhance the matching performance as (b) in Figure 3 is much more similar to Figure 4.

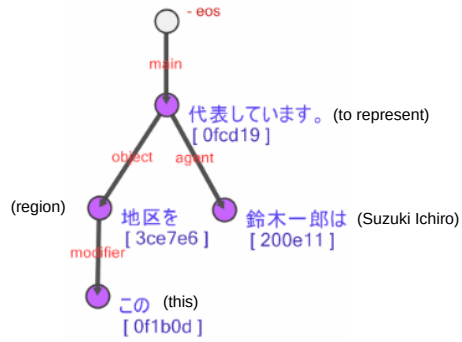


Figure 4. Semantic graph for the sentence
「鈴木一郎はこの地区を代表している。」
(Suzuki Ichiro represents this region.)

4. GRAPH MATCHING

As mentioned in Section 3.4, we start the matching process at the moment we succeed in locating the main predicate node in the graph of knowledge sentence. Then we visit the first adjacent node from the main predicate in the question sentence in a manner of depth-first search. Similarly, we also try to find the adjacent node of the main predicate in the knowledge sentence with the satisfaction that it matches the lately visited node in the question sentence. Repeat the above until all nodes in the question sentence have been visited, indicating the end of the matching process. In this section, we describe the matching procedure in detail.

4.1 Node matching

A node in the knowledge sentence must satisfy the following conditions to match a node in the question sentence.

Table 2. Matching rules for interrogative node

	node in question graph	node in knowledge graph
concept similarity	>= 0.5	
property	the same negation property	
POS	-	not verb
relation	relation similarity＋concept similarity >= 1.5	

Table 3. Matching rules for general node

	node in question graph	node in knowledge graph
concept similarity	>= 0.27	
referent	general concept	-
	proper concept	the same referent
property	the same negation property	
relation	relation similarity＋concept similarity >= 1.27	

Table 2 describes the case when the node in the question sentence is the interrogative node, and Table 3 is for the case of other nodes. Here, a concept similarity (denoted as CS below) is calculated by the following equation.

$$CS = \frac{2 \times dc(c_1, c_2)}{d(c_1) + d(c_2)}$$

In this equation, d(x) means the depth of the concept x in EDR's thesaurus, and dc(x, y) is the depth of the parent concept of x and y. Figure 5 is an example.

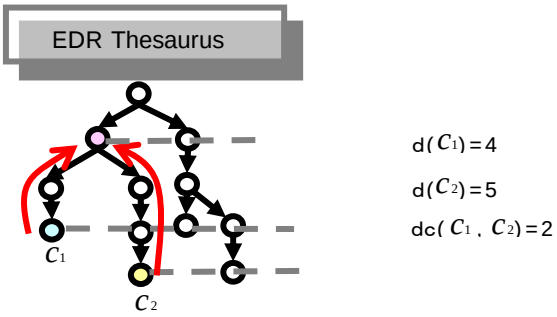


Figure 5. Depth of concepts in EDR thesaurus

In this way, the concept similarity between c_1 and c_2 is 0.44.

The concept of the interrogative node is replaced by the category concept as shown in Table 1 in section 3.1.

Another significant matching rule described in Table 2 and 3 is for the relation similarity. The relation similarity indicates the similarity between two relations: one in the question sentence and another in the knowledge sentence. A relation here means the semantic relation in the semantic graph from a parent node to its child node. In other words, besides the similarity of node pairs themselves, the relations they hold with their own children are also considered. It is easy to understand that the similarity will be 1 if the semantic relation between a pair of nodes in the question sentence is the same with the one in the knowledge sentence. How about the relations that are not the same? Table 4 shows the criteria for assigning similarities to relation pairs in this case.

Table 4. Similarities for relation pairs

relation similarity	deep case group
0.8	time,time-from,time-to
0.8	object,a-object,modifier
0.8	agent,object
0.8	a-object,modifier,possessor
0.6	goal,beneficiary,purpose
0.6	place,goal,from-to
0.4	place,goal,scene,a-object,modifier,from-to
0.4	cause,reason,logical
0.4	time,time-from,time-to,sequence,timing
0.4	manner,possessor,a-object,modifier

For instance, if the semantic relation is “agent” between a pair of nodes in the question graph, and “object” between the node pair being examined in the knowledge graph, we will assign the relation pairs with a similarity 0.8.

4.2 Node skipping

During the process described in section 4.1, not always could we go smoothly. Sometimes we are not able to continue our work due to the lack of matching nodes. The reason is variable including one that we might have just encountered some modification that is unconcerned, while the modified part following it is the node we are looking for. Here arises the necessity to skip one or more nodes in order to continue the matching work.

Node skipping takes place in both the question sentence and the knowledge sentence. Taking N_p as the visited node, N_c as the being examined node, i.e., the node to be skipped, and N_g as one of the children N_c holds, the algorithm will be as below.

foreach N_g *{*

generate a relation $N_p \rightarrow N_g$ *;*

$N_p \rightarrow N_g$ *= (* $N_p \rightarrow N_c$ *,* $N_c \rightarrow N_g$ *);*

DeepCase($N_p \rightarrow N_g$ *)= (DeepCase(* $N_p \rightarrow N_c$ *), DeepCase(* $N_c \rightarrow N_g$ *));*

delete $N_p \rightarrow N_c$ *and* $N_c \rightarrow N_g$ *;*

}

The new relation $N_p \rightarrow N_g$ is a list, and will bring a problem when matching another relation with it as described in the posterior half of section 4.1. In fact, we assign 1 to $N_p \rightarrow N_g$, if the deep case of $N_p \rightarrow N_c$ or N_c

$\rightarrow N_g$ exists for the other relation, and the similarity defined in Table 4 to $N_p \rightarrow N_g$, if $N_p \rightarrow N_c$ or $N_c \rightarrow N_g$ of another relation belongs to the same group in Table 4. Here are two examples showing the node skipping respectively in the question sentence and the knowledge sentence.

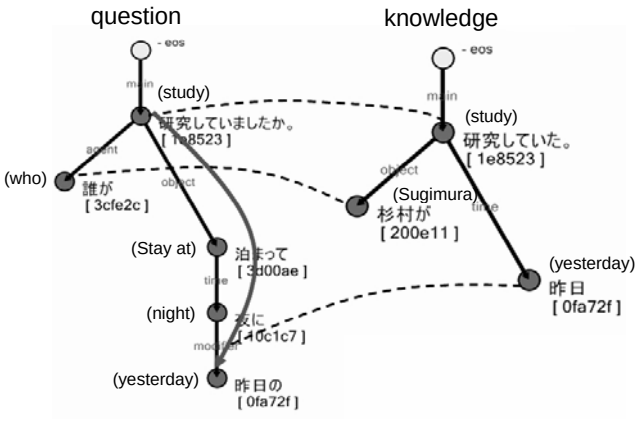


Figure 6. Node skipping in question sentence

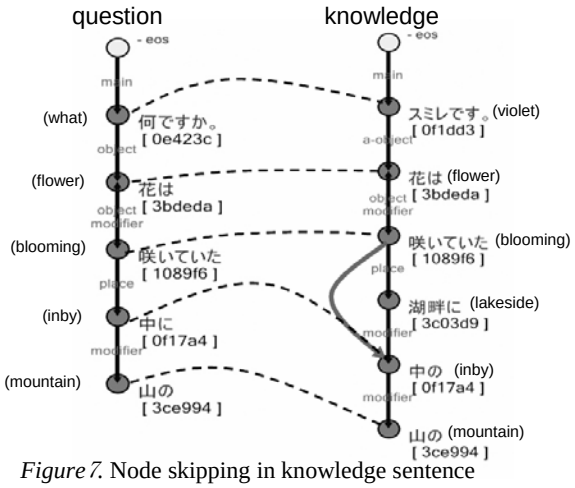


Figure 7. Node skipping in knowledge sentence

4.3 Calculation of graph similarities and answer extraction

Finally, after the above pcocedures, the node in the knowledge sentence matching the interrogative word in the question sentence is extracted as an answer candidate.

In case more than one answer candidate is available, similarities between the question graph and each knowledge graph are calculated as S_g and used to rank the answer candidates.

$$S_g = \frac{(S_n + S_a)}{2}$$

$$S_n = \frac{\sum N_{qk}}{N_q} \times 100$$

$$S_a = \frac{\sum A_{qk}}{A_q} \times 100$$

Here, N_{qk} represents the concept similarity of a pair of nodes, and N_q is the total number of nodes in the question sentence. Likewise, A_{qk} and A_q are for the Arcs, i.e., the relations..

5. EXPERIMENTAL RESULTS

We conducted an experiment to examine the effectiveness our method. Table 5 shows the results. Here we randomly extract 100 sets of question and answer from a TV quiz program [9] as the experimental data.

Table 5. Experimental results

		correct answer	wrong answer	no answer
suitable knowledge found (55%)	top 1	65%(36)	13%(7)	22%(12)
	till top 3	71%(39)	7%(4)	22%(12)
unsuitable knowledge found (45%)		0	16%(7)	84%(38)

Here the percentages in the correct answer column indicate the ratio of questions with correct answers found among all the 100 sets. Top 1 represents the ratio of questions for which the correct answers have been found by the topmost answer candidates, and Top 3 by one of the top 3 answers in the answer candidate lists.

We know from the table that suitable knowledge have been found for only 55% of the questions. This could probably be caused by the method we extract knowledge: although knowledge may exist across several sentences sometimes, we extract only an individual sentence.

Among the questions with suitable knowledge found, the success rate is 65%, and the failure rate 13%. The figures are not perfect, but prove the reliability of our method, especially for the low failure rate. We believe the delicate matching algorithm at the semantic level has reduced the wrong answers. And this could probably be the reason why 84% of the questions with no suitable knowledge found refused to provide answers.

6. CONCLUSION

The QA system we developed is effective in finding answers from Internet for given questions. We believe that we will get better performance with our system if we expand our method of extracting individual knowledge sentence to multiple ones. Also, it may be necessary to paraphrase the knowledge sentences, rather than the question sentences only as we do at present.

REFERENCES

1. Endo, T., and Fukumoto, J. : QA System using Classified Type of Named Entity. IPSJ SIG Notes. NL-159, pp.25-30 (2004).
2. Kurata, G., Okazaki, Naoki., and Ishizuka, Mitsuru. : Question Answering System with Graph Structure from Dependency Analysis. IPSJ SIG Notes. NL-158, pp.69-74 (2003).
3. Sasaki, Y., Isozaki, H., Suzuki, J., Kokuryou, K., Hirao, T., Kazawa, H., and Maeda, E. : SAIQA-II: A Trainable Japanese QA System with SVM (Natural Language Processing), Transactions of Information Processing Society of Japan. Vol.45, No. 2, pp.635-646 (2004).
4. Murata, M., Utiyama, M., and Isahara, H. : Question Answering System Using Similarity-Guided Reasoning. IPSJ SIG Notes. NL-135, pp181-188 (2000).
5. Maezawa, T., Menrai, M., Ueno, M., Han D., and Harada., M. : Improvement of the Precision of the Semantic Analysis System SAGE, and Generation of Conceptual Graph. Proceedings of the 66th National Convention of IPSJ, 2-6U-05, pp.177-178(2004).
6. Harada, M., Tabuchi, K., Oono, H. : Improvement of Speed and Accuracy of Japanese Semantic Analysis System SAGE and Its Accuracy Evaluation by Comparison with EDR Corpus. Transactions of Information Processing Society of Japan. Vol.43, No. 9, pp.2894-2902 (2002).
7. Harada, M., MIZUNO, T. : Japanese Semantic Analysis System SAGE using EDR. Transactions of the Japanese Society for Artificial Intelligence. Vol.16, No.1, pp.85-93 (2001).
8. Sowa, J. : Conceptual Structures, Information Processing in Mind and Machine, Addison-Wesley, Reading, MA (1984).
9. Quiz Millionaire, Fuji Television. (2002).