

MAIMAI: A Question Answering System at NTCIR3 QAC-1

Fumito MASUI, Masayuki MIYAGUCHI

Department of Information Engineering, Faculty of Engineering, Mie University
1515 Kamihama-cho, Tsu 514-8705, Japan
{masui,miyaguti}@shiino.info.mie-u.ac.jp

Abstract

This paper describes an question answering system based on syntactic information. Our system extracts answer candidates by ranking of score which shows similarity of syntactic structure. Syntactic structure is estimated based on answer type, density of weighty words, distance between words and depth of parse tree. To analyze syntactic structure, morphological analysis, named entity extraction and parser are utilized.

The system was used for runs of Task1 and Task2 on NTCIR3 QAC-1 and Experiments were conducted with the formal run test set. Results showed 0.157 mean reciprocal rank(MRR) for Task1 and 0.78 mean F value(MF) for Task2.

Keywords: *parse tree, similarity, named entity*

1 Introduction

Question Answering is a technique which extract an exact answer from an unstructured documents to a question expressed by natural language. Considering what kind of word can be detected as answer for question, we gave three assumptions for an answer below.

- An answer has a same type as the interrogative type or the type of word that explains an interrogative.
- It is possible that an answer is included in a sentence which has some important words appearing in question.
- It is possible that an answer is included in a sentence whose structure is similar with a question in a part.

According as the assumptions above, to find an answer, it is very important to analyze a sentence structure.

It can be said that there are some related work such as Murata's system [8], Collins' Tree Kernel method[1].

Murata proposed a method by which the answer is generated using similarity of parse tree based on transformation. In their method, main point is paraphrasing by transformation and only one level of syntactic relation is used.

The tree kernel method was explained with similarity of phrase structure based on number of common partial trees in a sentence. And Takahashi expanded Collins' method, to estimate syntactic similarity, some different measures can be explained with set of some parameters[10]. His method, however, evaluation is not realized for question answering task.

We implemented a question answering system which is based on similarity of syntactic structure. In our system, similarity is computed by some syntactic information from a maximum parse tree which includes important words.

To estimate an answer, our system utilizes some information based on syntax: type of the answer candidate(1), density of important word in a parse tree(2), and depth of a partial tree including an answer candidate and important words(3).

In this paper, first, overview of our Question Answering system is shown. Next, we conducted experiments to evaluate the performance of the system with QAC formal run test data[2]. At the end some topics of results are discussed.

2 Overview of MAIMAI System

In this section, we explains about the description of a question answering system "MAIMAI" which we implemented. Our system consists of three major modules which are *query analysis*(1), *document selection*(2) and *answer selection*(3) (see Figure1).

To generate an answer, the system retrieves documents with weighty words extracted from question and selects and ranks candidates for answer (Figure2).

In below, each module are explained.

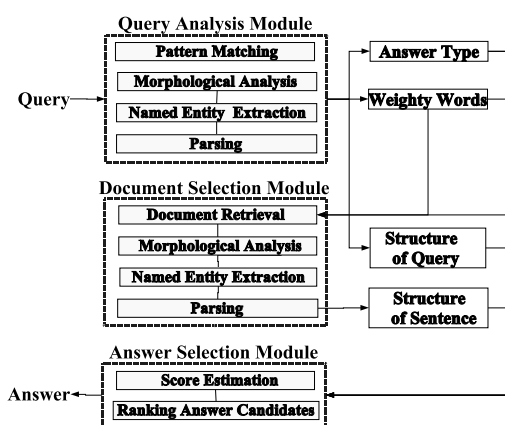


Figure 1. MAIMAI System

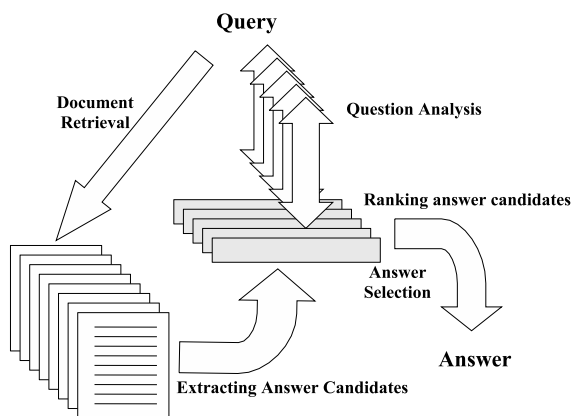


Figure 2. Question Answering Process

2.1 Query Analysis

In this module, an answer type for selection of sentences from which syntactic information is extracted and weighty words are extracted as query to retrieve documents. Weighty word set is extracted from parse tree of question with pattern matching rules written by regular expressions. The interrogative and its type are recognized for detection of the answer candidate from question. As a parser to get syntactic information, we used KNP[3] which runs on the output data from morphological analyzers Chasen[7] and JUMAN[4]¹.

¹ NExT[6] which is a NE Tagger converts the outline of JUMAN's morpheme to Chasen's one. Using NExT, we can unified two kinds of morphological analysis data.

Table 1. Samples of Template for Detection of Answer Type

pattern1	interrogative	pattern2	type
のは	だれ	ですか	PERSON
は	誰	でしょう	LOCATION /ORGANIZATION
*	どこ	でしたか	
	いつ	から	TIME
	どの*TYPE*	に	*TYPE*
	どこ*TYPE*	が	
	何*TYPE*	を	
	いくら	で	MONEY /NUMERIC
	どれくらい	*	ANY
	どれ		
	何なん		

Answer Type First, answer type is detected by matching templates up with the given question. With templates which is written by regular expressions and characters, the type related with the template matched up with question is regarded as the answer type. There are about 60 kinds of template which was created by human-handling to use in our system. We created about 20 templates for answer type detection. Some sample templates for pattern matching are shown in Table1. In the table, “pattern1” and “pattern2” means character part of template. And “interrogative” means the word indicates answer type. Answer type is shown in column “type,” “*TYPE*” which is decided with the word input permits any type of answer. Table1 shows six types for answer candidate: *PERSON*, *LOCATION*, *ORGANIZATION*, *TIME*, *MONEY* and *NUMERIC*. When it can not be detected which type the word has, the special type *ANY* is selected for the word.

For example, as Figure3, a question,

EX1. “ポパイの結婚相手は誰ですか。(To whom Popeye married?)”

is matched with a template such as “は+誰+ですか (whom)” and related type with the template “PERSON” is given as the answer type. Additionally, the other part of question “ポパイの結婚相手 (To X Popeye married)” is memorized and used for extraction of weighty word if it is needed to be used.

Weighty Word Above-mentioned weighty word should be the word that is essential to understand a question or the word which is important to find an answer from a sentence. Weighty word is, therefore, defined as an interrogative, the word having same type with interrogative (an answer candidate), the word related with interrogative in expression, and the word that is similar to the other weighty word. We realized

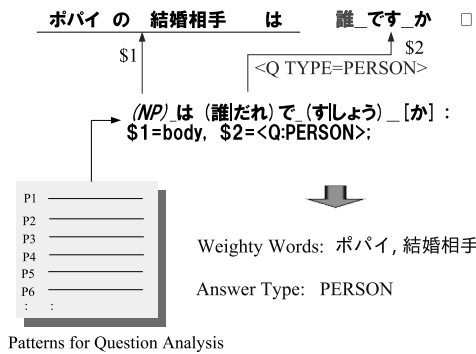


Figure 3. Detection of Answer Type and Weighty Words by Pattern Matching

detection of those words with *NEXT*[6]. And we used Chasen and JUMAN to produce morphological data required by *NEXT*.

As a weighty word, four kinds of words can be extracted. There are, in order of priority, “named entity,” “stem of a sa-hen verb or an adverb,” “noun,” and “suffix” as Table1 shows.

Syntactic Structure Detecting the sentence having a structure similar to a question, it is based on comparing structures of the question and structure of the sentence including weighty words. It can be realized to compare two structures with calculating similarity of structure between a sentence and a question. By the way, it is not useful that the structure of a whole sentence is compared because of high order calculation and incorrect parse tree from a complicated structure such as a compound sentence and a complex sentence.

Depth and Distance on Syntactic Structure Thereupon, we deal with similarity of partial trees in which weighty words are included. As measures for similarity of syntactic structure, two parameters, which are the structural distance between weighty word and the interrogative or the answer candidate(1) and the depth of maximum partial tree including the weighty word and the interrogative or answer candidate(2), are specified. Depending on those measurements, word order is free in the tree. Insertion of modifier is also permitted although it is possible that similarity of the tree become lower.

A value, which is the depth of maximum partial tree $dept(Root, W_i)$ and $dept(Root, Q)$. *Root* is the top node of the maximum partial tree. $dept(Root, Q)$ is

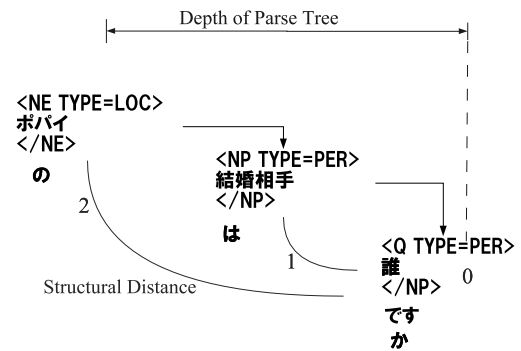


Figure 4. Detection of Depth and Distance of Parse Tree

total number of nodes between root node and the interrogative *Q*. $dept(Root, W_i)$ is total number of nodes between root node and each weighty word W_i . Getting parse tree, a parser KNP is utilized. If an interrogative can not be found in the partial tree, an empty interrogative(ϕ) is assumed as a terminal node.

Figure4 is an example of detection of tree depth and distance between weighty words. In Figure4, question word, “誰 (whom)” and, two weighty words, “ポパイ (Popeye)” as a *PERSON* type named entity, and “結婚相手 (To X Y married)” as a noun, are extracted from question(EX1). Additionally, three depths are computed utilizing parse tree of question. $dept(Root, 結婚相手)$ is determined as *one* and $dept(Root, ポパイ)$ is *two*. Because the node “誰” is a root node, in this case, $dept(Root, 誰)$ is regarded as *zero*.

2.2 Document Selection

In the document selection module, articles related with set of weighty words such as named entity and noun extracted from question are retrieved first. Articles including weighty words are retried from two year newspaper articles. To retrieval, the Namazu system ver.2.0.5[9] is utilized.

Next, the sentences including the weighty word whose type is matched with answer type are selected and parsed. With parse trees of those sentences, depth of parse tree and structural distance is calculated. There are some heuristic rules to solve problems. In the case of retrieving no article, it is tried to retrieve articles with the set of weighty words from which one word having the lowest priority are erased. In the case that some weighty words whose priorities are equal, the word that has the longest structural distance is

deleted. Additionally, a named entity is never deleted from set of weighty words because it is strongly possible to become an answer.

For the experiments, the retrieved articles ranked less than sixth were employed to be processed. To process selected sentences, Chasen/JUMAN, NExT, and KNP are also utilized same as query analysis.

2.3 Answer Selection

Answer selection also utilizes morphological analysis, named entity extraction and parsing to process retrieved documents. The words such as answer candidates and weighty words are extracted from the parse tree of the documents that is retrieved. Also, the word whose type is same as an interrogative in question is extracted as the answer candidate.

Estimation: Type of Weighty Word The weighted score, $w_1(A_i, W_j)$, is given to each weighty word that has same type with interrogative for its own priority. Named entity is given 40 because it is easy that this kind of word become an answer. The stem of sahen verb/adverb is given 15, other noun is given 10. From the reason that It is hard to be an answer itself with respect to suffix, parameter is not given. Value of each parameter is determined experimentally as expression(1) shows.

$$w_1(A_i, W_j) = \begin{cases} 40 & (if\ A_i/W_j\ is\ NE) \\ 15 & (if\ A_i/W_j\ is\ stem\ of \\ & \quad sahen\ verb/adverb) \\ 10 & (if\ A_i/W_j\ is\ noun) \end{cases} \quad (1)$$

If type of the candidate of answer is “named entity,” the $w_2(A_i)$ is given as 20 or if not, $w_2(A_i)$ is given as zero. The expressions to determine value of $w_2(A_i)$ is shown as expression(2).

$$w_2(A_i) = \begin{cases} 20 & (A_i\ is\ NE) \\ 0 & (otherwise) \end{cases} \quad (2)$$

Estimation: Depth of Parse Tree Assuming that candidate of answer that is one of the weighty words is A_i and the other weighty word is W_j , two kinds of values, $dept(Root, W_j)$, $dept(Root, A_i)$ and $dist(A_i, W_j)$ can be calculated.

Two kinds of depths can be computed with two word above. One is $dept(A_i, Root_{A_i})$, a distance between root and A_i , the other is $dept(W_j, Root_{A_i})$, a distance between root and W_j . In same way, two kinds of depth in question, $dept(Q, Root_Q)$ and $dept(W_j, Root_Q)$, are also determined already.

By the expression(1) parse tree of question and selected expression are compared. Adding two values,

which are computed by subtraction $dept(Root_{A_i}, A_i)$ from $dept(Root_Q, Q)$ and $dept(Root_{A_i}, W_j)$ from $dept(Root_Q, W_j)$, difference of two partial tree $diff(A_i, W_j)$ is computed.

$$diff(A_i, W_j) = (dept(Root_{A_i}, A_i) - dept(Root_Q, Q)) + (dept(Root_{A_i}, W_j) - dept(Root_Q, W_j)) \quad (3)$$

Moreover, $w_3(A_i, W_j)$ is decided experimentally as expression(4) shows. If $diff(A_i, W_j)$ is zero, which means two structures are different, $w_3(A_i, W_j)$ is given as zero. If $diff(A_i, W_j)$ is larger than zero, which means there is no difference between two structures, $w_3(A_i, W_j)$ is given as 50. $w_2(A_i, W_i)$ is given experimentally.

$$w_3(A_i, W_j) = \begin{cases} 0 & (diff(A_i, W_j) > 0) \\ 50 & (otherwise) \end{cases} \quad (4)$$

The $Score_0(A_i, W_j)$ is explained as sum of $w_3(A_i, W_j)$.

$$Score_0(A_i, W_j) = \sum_{j=0}^n w_3(A_i, W_j) \quad (5)$$

(6)

Estimation: Density of Weighty Words Density of weighty words in syntactic structure, $Score_1(A_i, W_j)$, is explained with structural distance and weight of word. Structural distance between an answer candidate A_i and a weighty word W_j is calculated by adding structure, $dept(Root_{A_i}, A_i)$ and $dept(Root_{A_i}, W_i)$. $Score_1(A_i, W_j)$ is determined by expression(7). If W_i appeared n times in article retrieved, the density of weighty words in a syntactic structure $Score(A_i, W_i)$ can be explained as expression(7).

$$Score_1(A_i, W_j) = \sum_{j=0}^n \frac{w_1(A_i, W_j)}{dept(Root_{A_i}, A_i) + dept(Root_{A_i}, W_j)} \quad (7)$$

Estimation: Ranking Answer Candidates To select an answer, $Score(A_i)$, that is total score of a answer candidate A_i , can be calculated with $Score_0(A_i)$ and $Score_1(A_i)$.

Finally, for Task1, answers candidate ranked in the top five in order of $Score(A_i)$ are output as system's answer.

$$Score_2(A_i, W_j) = Score_1(A_i, W_j) + Score_0(A_i, W_j) \quad (8)$$

$$Score(A_i) = Score_2(A_i, W_j) + w_2(A_j) \quad (9)$$

Table 2. Results of Evaluation in Task1 and Task2

	Question	Answer	Output	Correct	MRR	MF
Task1	200	288	725	45	0.157	—
Task2	200	288	679	45	—	0.078

MRR: Mean Reciprocal Rank MF: Mean F-value

For Task2, the answers which are same with the answers for Task1 are unified and output because the definition of Task2 requires that system should return all of answers.

3 Experiments

To learn our system’s performance, we conducted experiments with test sets of QAC formal run(Task1 and task2)[2]. Both test sets include 200 questions and their correct answer data. As measures, mean reciprocal rank(MRR) was used for task1 and Mean F-measure(MF) was used for task2.

The results of evaluation are shown in Table2. For task1, 50 of 725 system’s outputs were correct answer. The total performance of the system is 0.157 MRR points.

4 Discussion

Our participated system is implemented with hypothesis of running on Task1 mainly and testing on Task2 is optional. Consequently, we discuss about the system’s results of Task1 in this section.

Considering about total performance of our system in participated systems results, for task1, MRR ranks 12th out of 16 systems, MF ranks 8th out of 12 systems. This results means that there are some problems in the system implementation. First we discuss what kind of problem effected to the system performance.

It is hard to cause detection of an answer that morphological analysis can not succeed. In addition, false of named entity extraction, which are detection of named entity and its type, also cannot produce detection of an answer. Set of weighty words becomes keywords to retrieve articles. If a set of weighty words is not collect, incorrect article is retrieved. As a result, incorrect set of weighty words influences extraction of an answer.

False of extracting an important sentence brings about false of extraction an answer. It can be considered that we should have more investigating about an extraction of a weighty word and calculation of score.

Ranking answer candidates, as causes, it can be considered that computing depth of partial tree is influenced by performance of morphological analysis, named entity extraction and sentence structure analysis. In addition, parameters for scoring are determined

Table 4. Results of Correct Answers on Each Level

process level	population	correct	Recall
Question Analysis	200	92	0.41
Document Retrieval	92	51	0.53
Answer Selection	51	33	0.65

experimentally. It is deemed to be equipped with a theoretical model. Furthermore, estimating similarity of partial tree was too rigid to extract an answer candidate. It should be more flexible.

Table4 shows the results of performances from each process level. We picked up the results of query analysis, document retrieval and answer selection. In the table, only the cases in which the before process succeeded is used to consider, although there are some cases that answer type was not detected but selecting answer was succeeded (5 answers were included in the Table2).

Table4 shows that the answer selection performed 0.65 recall and the document selection shows 0.53 recall. “total” means that number of questions which the module could succeeded to process is counted as a population for the next process. In this case, the performance of document selection means just document retrieval’s result because the document including the correct answer type can be collected.

In the experiments, the configuration of retrieval system has not been set to be the best. As a consequence, it was false to retrieve correct articles for 51 questions. We certified that the Namazu system performs 0.66 precision for top five articles with the most suitable configuration to run. Using document selection with the best configuration, it can be considered that the number of our system’s correct answer will improve from 33 to maximum 40.

Question analysis shows only 0.41 recall. The main cause of faults in results is that we could not create enough templates to detect answer type. Now we are creating more templates to analyze a question and expand types of answer from 6 to 64. We believe that amount of templates and circumstantial answer types should effect to be more improved output of correct answers than above.

Analyzing relation between our system’s correct answers and their ranking position, the system ranked 26 words top(71.8%), 5 words second(10.3%), 4 words third(10.3%), 2 words fourth(5.1%) and one words fifth(2.6%). It can be said that our system’s performance becomes better for task2, if only top rank answers are output.

Table1 shows the result that correct answers of system’s output are clustered by answer type. Each type based on named entity extraction tool *NExT* is same with type used in query analysis. About three cases in

Table 3. Analysis of Answer Type and System's Performance in Task1

type	question	N.A.	1st	2nd	3rd	4th	5th	total	MRR	Recall
PERSON	44	1	4	1	1	1	0	7	0.118	0.163
LOCATION	30	1	6	0	1	1	1	9	0.234	0.138
ORGANIZATION	22	0	3	1	0	0	0	4	0.159	0.182
LOCATTION/ ORGANIZATION	5	1	2	1	1	0	0	4	0.708	1.000
MONEY	3	0	1	0	0	0	0	1	0.333	0.333
NUMERIC	25	1	1	1	1	0	0	3	0.076	0.125
TIME	17	0	3	0	0	0	0	3	0.176	0.176
ANY	54	1	6	1	0	0	0	7	0.123	0.132
total	200	5	26	5	4	2	1	38	0.157	0.195

*N.A.: no answer question which is ignored in Task1

Table 5. Results of System's Correct Answers on Each Rank in Task1

rank	correct	Ratio(%)
1st	26	71.8
2nd	5	10.3
3rd	4	10.3
4th	2	5.1
5th	1	2.6
total	38	100.0

which there is ambiguity for detection, the type "LOCATION/ORGANIZATION" is chosen exceptionally.

For LOCATION type, it is the best performance in Table3. Second, it is well-performed result for ORGANIZATION. Then it can be said that our system performed better for LOCATION type than other types. The words of PERSON and NUMERIC which appears as multi morphemes could not be extracted completely at named entity extraction process.

5 Conclusion

We described a question answering system based on structure analysis. Our system is extracted an answer by estimating similarity of syntactic structures between each sentence which includes an answer candidate and a given question. Estimating the similarity, depth of syntactic structure and density of weighty words are computed from parse tree of each sentence and a question.

Results of experiments in QAC-1 shows that MRR was 0.157 in Task1 and MF was 0.078 in Task2. Our system needs that much more templates to be utilized on query analysis. And the scoring algorithm is reconsidered theoretically.

Also the other QA system utilizing statistical ap-

proach is being implemented, to compare it with the participated system. Likewise, we developing a new system answering to more difficult question such as "What is AI?" based on a metaphor recognition model [5].

References

- [1] M. Collins and N. Duffy. Convolution kernels for natural language problems. Technical report, University of California at Santa Cruz, 2001.
- [2] J. Fukumoto, T. Kato, and F. Masui. Ntcir3 qac1 task home page. <http://www.nlp.cs.ritsumei.ac.jp/qac/>, 2001.
- [3] S. Kurohashi. Japanese parsing system knp manual(in japanese). 1998.
- [4] S. Kurohashi and M. Nagao. Japanese morphological analysis system JUMAN version 3.6 user's manual. 1998.
- [5] F. Masui, A. Morita, and J. Fukumot. An approach for question answering using metaphorical ground. IE-ICE Technical Report NLC2002-414-39, The Institute of Electronics, Information and Communication Engineering, 2002.
- [6] F. Masui, S. Suzuki, and J. Fkumoto. Next a named entity extraction tool. <http://irmscher.shiino.info.mie-u.ac.jp/next/>, 2001.
- [7] Y. Matsumoto, H. Kitauchi, T. Yamashita, Y. Hirano, H. Matsuda, K. Takaoka, and M. Asahara. User's manual for morphological analysis system "chasen" version 2.2.8. Naist technical report, Nara Advanced Institute Science and Technology, 2001.
- [8] M. Murata, M. Utiyama, and H. Isahara. Question answering system using similarity-guided reasoning. NI, Information Processing Society of Japan.
- [9] Namazu Project. Namazu: a full-text search engine. <http://www.namazu.org/>, 2002.
- [10] T. Takahashi, K. Inui, and Y. Matsumoto. Methods for estimating syntactic similarity. Technical report of ieice, The Institute of Electronics, Information and Communication Engineering, 2002.