Effect of combining different Web search engines on Web question-answering

Tatsunori Mori Akira Kanai Madoka Ishioroshi Mitsuru Sato
Graduate School of Environment and Information Sciences
Yokohama National University
79-7 Tokiwadai, Hodogaya-ku, Yokohama 240-8501, Japan
{mori,a-kanai,ishioroshi,mitsuru}@forest.eis.ynu.ac.jp

Abstract

We have several different commercial Web search engines that are available. It is expected that the combination of the results from different Web search engines has some impact on the accuracy of question-answering (QA) for Web documents, because the search results are not identical and the combination increases the variety of information source. However, as far as we know, there are no studies on the effect of combining different Web search engines on QA.

In this paper, we examined the effect of the combination on QA. We investigated three different methods to combine search results from different search engines in the process of QA. The first one is a conventional method that straightforwardly merges search results from different search engines, then, feeds the unified search result into one QA engine. On the other hand, the second one and third one are our proposal methods that feed each search result from individual search engine to a QA engine separately, then merge the answer candidates. The experimental result showed that the methods that merge the answer candidates after QA are more effective than the method that merges the search results before QA.

1 Introduction

The technology of question-answering (QA) is widely regarded as an advancement on the combination of information retrieval (IR) and information extraction (IE). QA systems do not provide us with the relevant documents; instead, they directly provide answers to questions. For example, when the system receives the question "What is the capital of Japan?", it searches 'knowledge resource' for answer candidates, and hopefully, it would return the answer "Tokyo."

With regard to knowledge resource, while earlier QA systems utilized static document collection such as a set of newspaper articles, many of recent studies focus on Web documents because the Web is an up-to-date information resource. We term the question-answering with Web documents *Web QA* in this paper. Since it is not realistic that QA services prepare their own Web crawler and Web search engine, existing commercial search engines are borrowed for Web QA systems. Fortunately, many of major search engine companies publicly offer their own APIs (application interfaces) to users.

Here, we would like to pay attention to the fact that we have *multiple different* Web search engines that are available for Web QA. It is expected that the combination of the results from different Web search engines has some impact on the accuracy of Web QA, because the search results are not identical and the combination increases the variety of information source.

However, as far as we know, there are no studies on the effect of combining different Web search engines on QA. In this paper, we examined the effect of the combination on QA. We will investigate three different methods to combine search results from different search engines in the process of QA. The first one is a conventional method that straightforwardly merges search results from different search engines, then, feeds the unified search result into one QA engine. On the other hand, the second one and third one are methods that feed each search result from individual search engine to a QA engine separately, then merge the answer candidates.

2 Related work

Lin et al.(Lin and Katz, 2003) give us a good tutorial about Web QA. According to it, there are at least two ways of using Web documents. First one is to use Web documents as the primary corpus of information. Second one is to combine use of Web documents with other corpora. In this paper, we focus on the first way of use.

It is pointed out that one advantage of use of Web document is the data redundancy of Web documents. The expressiveness of natural language allows us to say the same thing in multiple ways. The fact is usually one main problem of question answering, because an answer may be stated in different ways from a question. However, with data redundancy on the Web, it is likely that the answer will be stated in the same way as the question was asked(Lin and Katz, 2003), because a lot of different authors may describe the answer in their own ways. That is, the data redundancy gives us the variety of description.

From the viewpoint of variety of description, several researches take advantage of variety of information source. For example, recent version of START, which is one of the first Web-based QA system, makes use of multiple information source(Katz et al., 2004, 2005). Radev et al. proposed a probabilistic approach to question answering on the Web(Radev et al., 2005), and they uses three major search engines to retrieve the top 40 documents.

Although these researches utilize documents from different information sources, they make no distinction between information sources after document retrieval. On the other hand, our methods described in Section 4 try to exploit the data redundancy among the difference information sources.

3 Basic question-answering system

The basic QA system used in this study is a real-time QA system based on the study by Mori (Mori, 2005). It can answer Japanese factoid questions in Japanese. As shown in Figure 1, the system comprises six parts of process — the question analysis, interface to external search engine, passage extraction, sentential matching, answer generation, and pseudo voting.

The process of question analysis receives a question from a user and extracts several kinds of information including a list of keywords, the question type, and so on. Here, we define the term *Keywords* as content words in a given question. The list of keywords is submitted to an external search engine to retrieve relevant documents.

The process of sentential matching receives a set of sentences from the passage extractor. It treats each morpheme as an answer candidate and assigns it a matching score as described below. It should be noted that a morpheme may be either a word or a part of a longer compound word. Therefore, in the latter case, the process of answer generation extracts the compound word including the answer candidate, and treats it as a proper answer candidate.

3.1 Raw scores for answer candidates

In the basic QA system, a composite matching score for an answer candidate is adopted as shown in Equation (1). We term it *raw score* in this paper. It is a linear combination of the following sub-scores for an answer candidate AC in the i-th retrieved sentence L_i with respect to a question sentence L_q:

1. $Sb(AC, L_i, L_q)$, the matching score in terms of character 2-grams;
2. $Sk(AC, L_i, L_q)$, the matching score in terms of the keywords;
3. $Sd(AC, L_i, L_q)$, the matching score in terms of the dependency relations between an answer candidate and the keywords; and
4. $St(AC, L_i, L_q)$, the matching score in terms of the question type.

In the calculation of $St(AC, L_i, L_q)$, we employ an NE recognizer that identifies eight types of NEs defined in the IREX-NE task(IREX Committee, 1999).

$$\begin{aligned} S(AC, L_i, L_q) = \\ & Sb(AC, L_i, L_q) + Sk(AC, L_i, L_q) + \\ & Sd(AC, L_i, L_q) + St(AC, L_i, L_q) \end{aligned} \quad (1)$$

In order to reduce the computational cost, the A* search control is introduced in the sentential matching mechanism. With this control, the system can process the most promising candidate first, while delaying the processing of the other candidates, and perform the n-best search for the answer candidates.

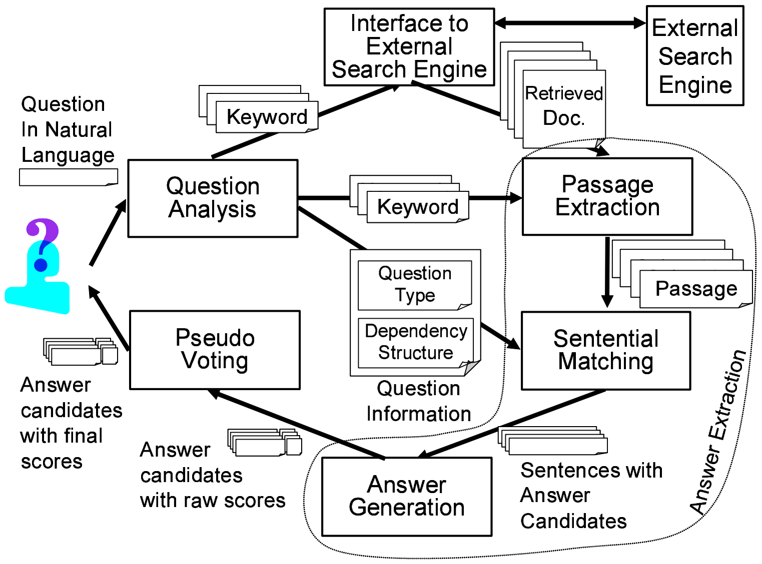

Figure 1: Basic QA system

3.2 Pseudo voting method in search scheme

Many existing QA systems exploit global information on answer candidates. In particular, redundancy is the most basic and important information. For example, there are previous studies that boost the score for answer candidates that occur multiple times in documents(Clarke et al., 2001; Xu et al., 2003). This is known as the *voting method*.

On the other hand, we cannot exploit the voting method directly in the scheme of searching answers because the system quits the searching after n-best answers are found. Therefore, an approximation of the voting method, termed *pseudo voting*, is introduced as follows. In case n-best answers are necessary, the system continues searching for answers until *n different* answer candidates are found. Therefore, the system may find other answer candidates that have the same surface expression as one of the answer candidates that have already reached the goal state of search. Consequently, we can partially use the frequency information of answer candidates by recording all that have reached the goal state in the search process. In this paper, the pseudo voting score $S^v(AC, L_q)$ for an answer candidate AC is defined as follows:

$$S^v(AC, L_q) = (\log_{10}(freq(AC, AnsList)) + 1) \cdot \max_{L_i} S(AC, L_i, L_q) \quad (2)$$

where *AnsList* is the list of answer candidates that have reached the goal state in the n-best search, and $freq(x, L)$ is the frequency of x in L. We also term the pseudo voting score *the final score* in this paper.

According to the experiments by Murata et al.(Murata et al., 2005), the above voting score is comparable with other good voting scores.

3.3 Exploiting Web documents by using snippets in Web search results

By replacing the document search engine with a Web search engine, the basic QA system can be easily exploit Web documents. However, downloading a couple of hundred Web documents is time-consuming task. To address the problem, Sagara et al.(Sagara et al., 2006) empirically showed that the use of snippets, which are short extractive summary produced by Web search engine, are effective as the resource for Web QA. Katz et al.(Katz et al., 2005) also use the snippets to generate answer candidates. We follow the same line in order to reduce the turn-around time.

Figure 2 shows the structure of the basic Web QA system, which utilizes the snippets from a search engine and has the basic QA system described above as a QA engine. In this figure, “the wrapper program for Web Search Engine” is a program component that absorbs the difference of protocols between each API of a particular search engine and the basic QA engine. It first receives a query from the basic QA engine, and then submits the list of keywords as the query to a particular search engine. Second, it extracts snippets from the search result and returns the snippets to the ba-

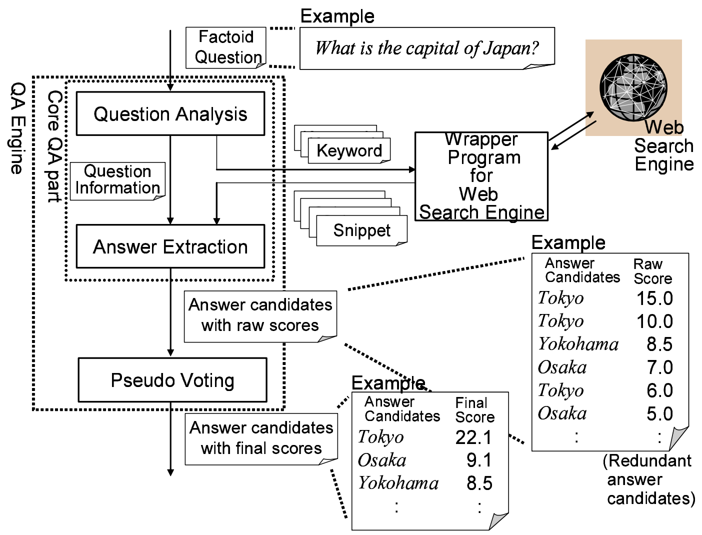

Figure 2: Basic Web QA system

sic QA engine as a set of documents. Finally, the basic QA engine performs question-answering.

4 Methods to combine search results from different Web search engines

There are, at least, the following three methods to combine the results from different Web search engines.

Combination A: the method that straightforwardly merges search results from different search engines, then, feeds the unified search result into one QA engine.

Combination B: the method that feeds each search result from individual search engine to a QA engine separately, and then merges the answer candidates with raw scores before pseudo voting.

Combination C: the method that feeds each search result from individual search engine to a QA engine separately, and then merges the answer candidates after pseudo voting. Since here we may have redundant answer candidate, the second voting is performed with Equation (2). Here, 'the raw score' for the second voting is the final score after the first pseudo voting.

Figures 3, 4 and 5 show the combination A, B, and C, respectively. In these figures, a "merger" receives lists of data from multiple sources and just merges the lists.

The combination A is the baseline method, which merges the result from different search engines *before* the question-answering process. The same kind of approaches is adopted in other researches as described in Section 2. The combinations B and C are our proposed methods, which merge the results from different search engines not before the question-answering but *after* the core question-answering process.

It should be noted that Combination B and C are different from Combination A from the viewpoint of the variety of information source. Since both Combination A and Combination B extract answer candidates with higher raw scores from snippets produced by Web search engines, they may be supposed to produce the almost same answer candidates. However, in Combination A, all answer candidates are treated equally in the ranking regardless of the difference of information source. Therefore, the answer candidates that are forwarded to the pseudo voting process may be resulted only from a small number of search engines. On the other hand, in Combination B, the answer candidates to be voted come evenly from all search engines. Thus, the variety of information source is ensured. Combination C is also in the same situation as Combination B. Moreover, in Combination C, answer candidates from many different information source would receive higher score in the stage of the second voting.

Here, we have a hypothesis that the certainty of answer candidates can be measured by the variety of information sources. If the hypothesis holds, we

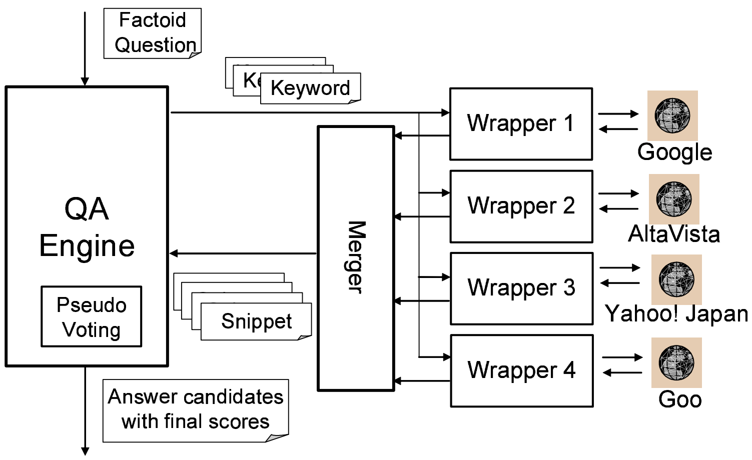

Figure 3: Baseline QA system with multiple search engines (Combination A)

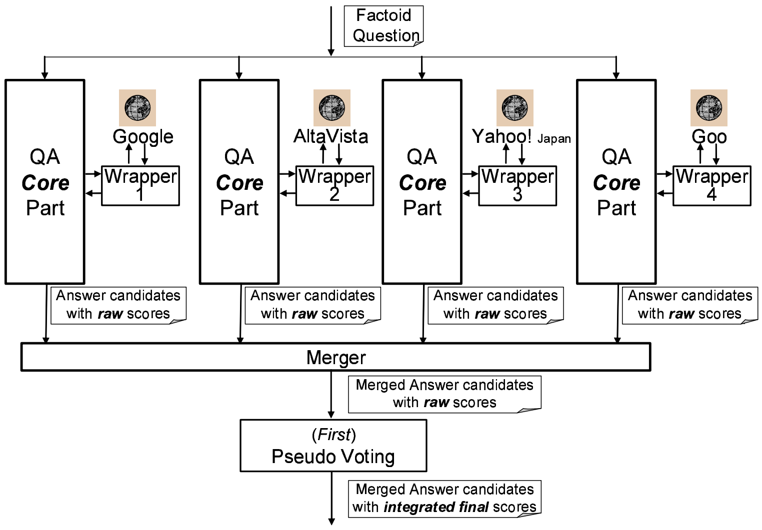

Figure 4: QA system with multiple search engines (Combination B)

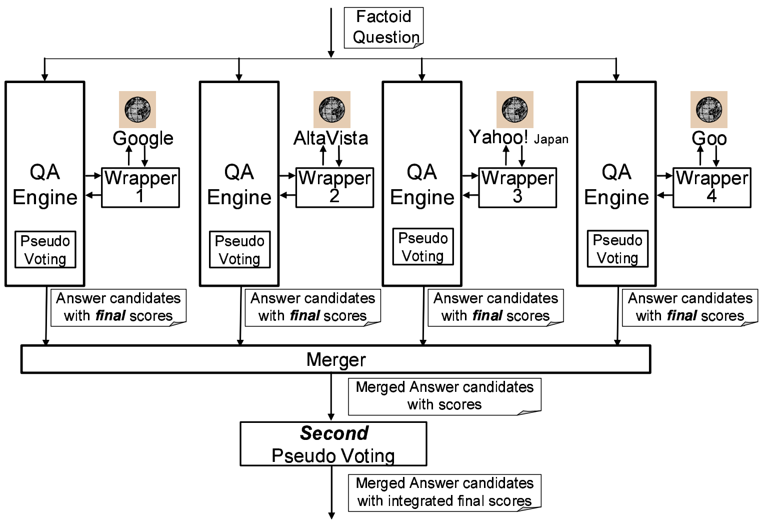

Figure 5: QA system with multiple search engines (Combination C)

may expect that Combination B and C are more accurate than Combination A.

5 Experimental result

In order to evaluate three combination methods, we conducted question-answering experiments as described below. In the same condition, we also examine the accuracy of the systems that use only one of the search engines as baselines.

5.1 Question set and their answer set

As for the question set and their answer set, we use a subset of the question set of NTCIR-3 QAC1(Fukumoto et al., 2002) and their corresponding official answer strings. NTCIR QAC is a series of evaluation workshops for question-answering organized by the National Institute of Informatics, Japan. The question set of NTCIR-3 QAC1, which consists of factoid-type questions, is originally designed for evaluating Japanese QA systems that have Japanese newspaper articles in 1998 and 1999. Therefore, the *current* answers of some questions have been changed from the official answers[1]. However, we would not like to judge the answer candidate with our own criterion, because of the neutrality and objectivity of the evaluation. Therefore, we just skipped this sort of questions.

The other issue is the problem of hanging-up of APIs. It is very rare, but some APIs of Web search engines we used hung up by the merest chance. In this evaluation, when at least one API timed out, we also skip the question.

By skipping these sort of problematic questions, from the question set of QAC1, we selected 50 questions that have earlier ID numbers. The actual question set is shown in the appendix as a list of question IDs.

5.2 Other experimental settings

With regard to Web search engines, we made use of the following Japanese-capable Web search engines as external search engines:

1. Google (`http://www.google.co.jp/`, Google SOAP Search API),
2. goo (`http://www.goo.ne.jp/`),
3. AltaVista (`http://www.altavista.com/`),
4. Yahoo! Japan (`http://www.yahoo.co.jp/`, Yahoo! JAPAN Developer Network).

All search results were obtained in February 24th and 25th, 2007.

As for the parameters related to the QA engine, the setting shown in Table 1 is used. According to Mori (Mori, 2005), the combination of the values is effective, at least, when the information source is a set of newspaper articles.

Here, it should be noted that the total number of documents for the systems of the combination methods is four times larger than the systems with only one of search engines, because the combination methods are utilize four different search engines with the same value of the parameter d. Although it is a very difficult problem to tell what is the fair comparison, one other possible choice of setting would be that the number of documents for the methods with only one search engine is increased four times. However, Mori (Mori, 2005) reported that increasing the number of documents does not necessarily improve the accuracy of question-answering because the lower ranked documents may eventually yield incorrect answer candidates that have lower raw scores but have high frequencies. Finally, we straightforwardly adopted the original setting of the parameters in Table 1.

5.3 Results

The accuracy of each combination method was evaluated using the mean reciprocal rank (MRR) and the average accuracy of the top n answer candidates. Reciprocal rank (RR) is the inverse of the rank of the first correct answer for each question. If no correct answer appears within the top five answer candidates, RR is 0. MRR is the average of RR over all questions. On the other hand, "Top n" represents the ratio of the number of questions whose one correct answer is found within the top n answer candidates.

The evaluation results are summarized in Tables 2 and 3.

6 Discussion

As shown in Table 2, the combination methods outperform methods that utilize only one Web search engine. Combination B and C in particular have good performance in terms of accuracy.

[1]For example, the answer to the question "What is the population of Tunisia?" (the question ID is QAC1-1019-01 and it is originally written in Japanese.) should vary year by year when we use Web documents as information source.

Table 1: Parameter setting of the QA engine for experiments

Parameter name	Value	Description
a	10	Number of answers to be searched.
d	250	Number of documents (snippets) to be retrieved.
ppd	5	Maximum number of passages retrieved from one document (snippet).
p	30	Number of passages to be considered in the retrieved documents (snippets).

Table 2: Accuracy of each combination methods

		Accuracy				
Method	MRR	Top 1	Top 2	Top 3	Top 4	Top 5
Google only	0.349	0.280 (14/50)	0.360 (18/50)	0.420 (21/50)	0.440 (22/50)	0.460 (23/50)
Goo only	0.314	0.220 (11/50)	0.300 (15/50)	0.420 (21/50)	0.260 (23/50)	0.480 (24/50)
AltaVista only	0.356	0.300 (15/50)	0.380 (19/50)	0.400 (20/50)	0.420 (21/50)	0.440 (22/50)
Yahoo! Japan only	0.376	0.300 (15/50)	0.420 (21/50)	0.440 (22/50)	0.460 (23/50)	0.480 (24/50)
Combination A	0.392	0.320 (16/50)	0.400 (20/50)	0.480 (24/50)	0.500 (25/50)	0.500 (25/50)
Combination B	**0.456**	**0.360** (18/50)	**0.480**(24/50)	**0.560** (28/50)	**0.580** (29/50)	**0.600** (30/50)
Combination C	0.451	**0.360** (18/50)	**0.480** (24/50)	0.520 (26/50)	0.560 (28/50)	**0.600** (30/50)

Although Combination B was well performed than Combination C, the difference is not so significant.

The result shows that the methods that merge the answer candidates after question answering is more effective than the method that merge the search results before question answering. As described in Section 4, in the former methods, i.e. Combination B and C, the answer candidates to be voted come evenly from all search engines, and the variety of information source is ensured.

According to Table 3, there are 33 questions for which at least one system in Table 2 finds correct answers within Top 5. While the systems with only one search engine found correct answers only for 22-to-24 questions within Top 5, Combination B and C improved the accuracy and could correctly answer 30 questions. From the fact, we may conclude that answer candidates from different search engines can complete each other. The fact also appears to support our hypothesis described in Section 4, namely, the hypothesis that the certainty of answer candidates can be measured by the variety of information sources.

7 Conclusion

We examined the effect of combining different Web search engines on QA. We investigated three different methods to combine search results from different search engines in the process of QA. The experimental result showed that the methods that merge the answer candidates after QA are more effective than the method that merges the search results before QA.

In this paper, we investigated only three primitive combination methods. The development of more effective combination method should be included in our future work.

Acknowledgements

We are grateful to the task organizers of NTCIR-3 QAC1 and people who manage the NTCIR workshops. We would like to especially thank Mainichi Shimbun and Yomiuri Shimbun for permitting us to use the documents for research. We would also like to thank the anonymous reviewers for their helpful comments.

This study was partially supported by Grant-in-Aid for Scientific Research on Priority Areas (No.18049031 and No.19024033) from the Ministry of Education, Culture, Sports, Science and Technology, Japan.

References

Charles L.A. Clarke, Gordon V. Cormack, and Thomas R. Lynam. 2001. Exploiting redundancy in question answering. In *Proceedings of SIGIR '01: the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 358–365.

Jun'ichi Fukumoto, Tsuneaki Kato, and Fumito Masui. 2002. Question Answering Challenge (QAC-1) — Question answering evaluation at NTCIR Workshop 3 —. In *Working Notes of the Third NTCIR Workshop meeting – Part IV: Question Answering Challenge (QAC1)*, pages 1–6.

IREX Committee, editor. 1999. *Proceedings of IREX workshop*. IREX Committee. (in Japanese).

Table 3: Distribution of questions in terms of difficulty in answering

Number of questions for which at least one system in Table 2 finds correct answers within Top 5	33/50
Number of questions for which all systems in Table 2 find correct answers at Top 1	7/50

B. Katz, M. Bilotti, S. Felshin, A. Fernandes, W. Hildebrandt, R. Katzir, J. Lin, D. Loreto, G. Marton, F. Mora, and O. Uzuner. 2004. Answering multiple questions on a topic from heterogeneous resources. In *Proceedings of TREC 2004*.

B. Katz, G. Marton, G. Borchardt, A. Brownell, S. Felshin, D. Loreto, J. Louis-Rosenberg, B. Lu, F. Mora, S. Stiller, O. Uzuner, and A. Wilcox. 2005. External knowledge sources for question answering. In *Proceedings of TREC 2005*.

Jimmy Lin and Boris Katz. 2003. Question answering techniques for the world wide web. URL `http://acl.ldc.upenn.edu/eacl2003/papers/tutorial/t2.pdf`, (Tutorial presentation at The 11th Conference of EACL (EACL-2003)).

Tatsunori Mori. 2005. Japanese question-answering system using A* search and its improvement. *ACM Transactions on Asian Language Information Processing (TALIP)*, 4(3):280–304. URL `http://portal.acm.org/TALIP/`.

Masaki Murata, Masao Utiyama, and Hitoshi Isahara. 2005. Use of multiple documents as evidence with decreased adding in a japanese question-answering system. *Journal of Natural Language Processing*, 12(2):209–247.

Dragomir R. Radev, Weiguo Fan, Hong Qi, Harris Wu, and Amardeep Grewal. 2005. Probabilistic question answering on the web. *Journal of the American Society for Information Science and Technology*, 56(3).

Haruki Sagara, Tatsunori Mori, and Hiroshi Nakagawa. 2006. Effectiveness of snippets of web search engine as a knowledge source for qa. In *Proceedings of the twelfth annual meeting of the Association for Natural Language Processing*, pages 316–319. (in Japanese).

Jinxi Xu, Ana Licuanan, and Ralph Weischedel. 2003. TREC2003 QA at BBN: Answering definitional questions. In *Proceedings of the twelfth Text Retrieval Conferene (TREC 2003)*.

A Questions used in the experiment

We use the following 50 questions from NTCIR-3 QAC1. The full set of questions is available at the online proceedings of NTCIR Workshop 3 (Formal Run Questions of Task 1 in Question Answering Task, `http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings3/`):

QAC1-1006-01, QAC1-1009-01, QAC1-1010-01, QAC1-1016-01, QAC1-1017-01, QAC1-1020-01, QAC1-1021-01, QAC1-1022-01, QAC1-1023-01, QAC1-1024-01, QAC1-1025-01, QAC1-1026-01, QAC1-1027-01, QAC1-1028-01, QAC1-1029-01, QAC1-1030-01, QAC1-1031-01, QAC1-1032-01, QAC1-1033-01, QAC1-1034-01, QAC1-1035-01, QAC1-1036-01, QAC1-1037-01, QAC1-1038-01, QAC1-1039-01, QAC1-1040-01, QAC1-1041-01, QAC1-1042-01, QAC1-1043-01, QAC1-1045-01, QAC1-1046-01, QAC1-1047-01, QAC1-1048-01, QAC1-1050-01, QAC1-1051-01, QAC1-1054-01, QAC1-1055-01, QAC1-1056-01, QAC1-1057-01, QAC1-1058-01, QAC1-1059-01, QAC1-1060-01, QAC1-1064-01, QAC1-1066-01, QAC1-1067-01, QAC1-1068-01, QAC1-1069-01, QAC1-1070-01, QAC1-1071-01, QAC1-1073-01.