# Question Answering System for Entrance Exams in QA4MRE

Xinjian Li, Tian Ran, Ngan L.T. Nguyen, Yusuke Miyao, and Akiko Aizawa

National Institute of Informatics, Tokyo, Japan
{lixinjian,tianran,ngan,yusuke,aizawa}@nii.ac.jp

**Abstract.** This paper describes our question answering system for Entrance Exams, which is a pilot task of the Question Answering for Machine Reading Evaluation at Conference and Labs of the Evaluation Forum (CLEF) 2013. We conducted experiments in which participants were provided with documents and multiple-choice questions. Their goals was to select one answer or leave it unanswered for each question. In our system, we developed a component to detect all story characters in the documents and tag all personal pronouns using coreference resolution. For each question, we extracted related sentences and combined them with candidate answers to create inputs for a Recognizing Textual Entailment (RTE) component. The answers were then selected based on the confidence scores from the Recognizing Textual Entailment component. We submitted five runs in the task and the run that ranked highest obtained a c@1 score of 0.35, which outperformed the baseline c@1 score of 0.25.

**Keywords:** Question Answering Systems, Coreference Resolution, Recognizing Textual Entailment

## 1 Introduction

Question Answering for Machine Reading Evaluation (QA4MRE) is a lab that has been offered in Conference and Labs of the Evaluation Forum (CLEF) since 2011. It is an exercise to develop a methodology for evaluating machine reading systems through Question Answering and Reading Comprehension Tests [1]. In the lab, participants are provided with several documents and questions, the answer for each question is to be stated or implied in the document. The goal for participants is to develop a system to extract corresponding knowledge from the documents, thereby solving questions with them. The main task is composed of four topics, Aids, Climate Change, Music and Society, and Alzheimer. Besides the main task, QA4MRE at CLEF 2013 also offers two pilot tasks; Machine Reading of Biomedical Texts about Alzheimer's Disease and Entrance Exams. Our discussion mainly focuses on our participation in Entrance Exams.

Entrances Exams is a new task for evaluating machine reading systems by solving problems from Japanese university entrance exams. Similar tasks using Japanese entrance exams were also held in NTCIR RITE [4]. The challenge for

this task is to test systems in the same situation in which high school students are evaluated. While a diverse range of background knowledge, such as Wikipedia entries, are available to all participants in other tasks, no external sources are provided in this task; therefore, systems are expected to make use of a high-school level of common sense. Only reading comprehension exercises were included in QA4MRE at CLEF 2013. Other types of exercises will be used in the future.

The rest of this paper is composed as follows: Section 2 describes the details of system's architecture. Section3 presents the results of our participation and their evaluations, and Section 4 concludes the paper.

## 2 System Architecture

In this section, we describe the detailed architecture of our system in the Entrance Exams task. Although the task was offered for the first time in the lab, the basic approaches do not significantly differ from other tasks. We referred to studies attending other tasks in previous QA4MRE tasks to develop our system [2][3]. Our system consists of three components, as shown in Figure 1.

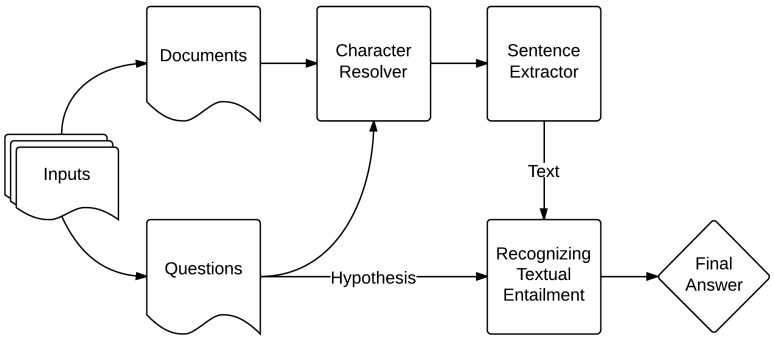


**Fig. 1.** System Architecture

The first component is the *Character Resolver*, which detects all story characters appearing in the documents and applies coreference resolution to personal

pronouns. The second component is the *Sentence Extractor*. Questions are classified into several types in this component then related sentences are extracted for each question from the document. The last component is Recognizing Textual Entailment (RTE), in which we calculate the most likely answer for each question. The following subsections describe the details of these three components. Besides these three main components, a component was also developed to process documents and questions with a parser [5]. Since this is just a simple preprocessor, we do not discuss it in this paper.

### 2.1 Character Resolver

Given an actual exam-data from QA4MRE, we found the documents for entrance examinations were mainly composed of stories and related questions. We also found that most of the questions were connected with actions of story characters in these documents. To answer each question correctly, therefore, it is important to detect which character is responsible for the focused upon action. The *Character Resolver* was developed to detect all story characters and mentions of them in the documents, including coreferential mentions. To achieve this, we divided the task into two small processes as shown in the following pseudo code. The first one is for detecting all nouns for characters and to merge them into groups if their mentioned characters are identical. In the second process, we unify personal pronouns into the same person groups.

*Pseudo Code for Character Resolver*

```
Initialize personDict to an empty hash table
Initialize personCount to 0
Initialize personGroups to a empty list

Process 1 :
  For each sentence in documents and questions
    For each word in sentence
      If word is detected by Name Entity Recognizer or
      If word matches prepared list
        If word is in personDict
           wordID := personDict [word]
        Else
           personCount := personCount + 1
           personDict[word] := personCount
           wordID := personCount
           Create a newGroup for wordID
           Add the newGroup to personGroups

  For each personGroup1, personGroup2 in personGroups
     If personGroup1 and personGroup2 contain the same person
```

```
      Merge personGroup1 and personGroup2

Process 2 :
  For each sentence in documents, questions
    For each word in sentence
      If word is a personal pronoun
        wordID := coreference_resolution (word)
        Find personGroup1 containing wordID
        Add word to personGroup1
```

In Process 1, we combine a prepared words list with the Name Entity Recognizer from Stanford [6]. First, the documents are divided into sentences and each sentence is divided into words. Then we use the Name Entity Recognizer to find proper names in the documents. Since the Name Entity Recognizer only detects nouns that seem to be names, it fails to detect common nouns that indicate general characters such as "teachers" and "mother". To address this problem, we prepare a list consisting of character reference nouns and search the documents to determine whether these words appear. Each word or name in the text is then assigned a person ID for ease of management. In most cases, several detected nouns might denote the same person, which would cause bad influence in the Recognizing Textual Entailment component. Therefore, we also clustered some words with the same meaning such as "father" and "dad". For example, "mother" and "mom" in the following text are tagged as the same person.

*Her* **<coref id="2">** *mother* **</coref>** *must have heard the front door close.* **<coref id="1">** *Christine* **</coref>** *went in and sat on the sofa. "How was your exam, dear?", her* **<coref id="2">** *mom* **</coref>** *asked.*

After character detection, we classify personal pronouns in the documents and questions. We develop a tool for this task by just following simple rules such as plural and male/female. The most useful feature of this tool for this task is that we can to some extent successfully tag words like "I" and "you" from conversations in the text. General coreference resolution tools do not perform well in this respect, but this feature is important in the Entrance Exams task because conversations might occupy a large percentage of the text and it would be useful to tag these pronoun words. After this application, we generate a tagged text like the following conversation.

*The* **<coref id="3">** *professor* **</coref>** *looked at* **<coref id="2">** *me* **</coref>** *and smiled. "Oh!* **<coref id="2">** *You* **</coref>** *speak Japanese very well."*

### 2.2 Sentence Extractor

Since the final goal with our system is to select the correct answer based on the output from the Recognizing Textual Entailment component. Both *Text* and *Hypothesis* should be generated as input to it. The "*Text* entails *Hypothesis*" means that a person would determine the *Hypothesis* as almost correct after reading the *Text*. In this component, we extract related sentences from documents. This is achieved by giving sentences *relevant scores* for each question. Sentences ranked highest are those we should extract.

Before the extraction process, all questions are classified into six types based on the interrogative as the following table shows. The classification indicates which sentences we should extract. We can prepare a list of keywords for each question type. The sentences that contain these expressions are more likely the sentences we need. For instance, related sentences corresponding to "Why" questions could possibly include a word such as "because". Preparing all expressions is impossible, so we mainly depend on the Name Entity Recognizer to find expressions for LOCATION, PERSON and DATE. We also does not prepare a list for questions such as "What" questions, because it is too vague to prepare a list of relevant words.

**Table 1.** Question Types

| Types | Interrogatives | Relevant expressions |
|---|---|---|
| OBJECT | What, Which | N/A |
| LOCATION | Where, Which location | city, country, village ... |
| PERSON | Who, Which person | Tom, James ... |
| DATE | When, What time | day, year ... |
| REASON | Why | because of, due to ... |
| Others | How ... | N/A |

For each keyword we add five point to the *relevant score* of sentences containing them. Then, we measure the similarity between questions and sentences. First, each word in documents and questions is assigned a tf-idf weight which we calculated from the Wikipedia corpus in advance. Second, the following features are combined to add as the *relevant score*.

a **Word similarity.** For each sentence, we calculate its similarity with a question. This similarity is determined by word similarity using WordNet. This feature would add at most 20 points to each sentence.
b **Dependency similarity.** In the preprocess for documents and questions, dependencies are also generated by the parser [8]. One point is added as the *relevant score* if the same dependency appears in both a sentence and a question.

c **Character reference.** Every story character in the documents is assigned an ID number in the previous component. We add five points to sentences if the same person is referred.

With scores computed, all sentences are ranked for each question and the sentence that has the highest score is extracted. Except for the highest one, the sentences surrounding the highest one might also contain important information for the question. Therefore, we extracted five sentences in our experiments, the highest one and four sentences surrounding it. These sentences were used as the *Text* in our Recognizing Textual Entailment system.

### 2.3 Recognizing Textual Entailment

Our Recognizing Textual Entailment component uses semantically annotated dependency parses of sentences as logical representations, utilizing an inference engine to perform logical inferences on such representation. The knowledge it uses in the inference process is obtained from synonym/antonym/hypernym relations in WordNet. Furthermore, it also uses an abduction component to generate alignments between small pieces of the Text/Hypothesis pair (T/H pair), which corresponds to may-be-missing knowledge that can make the logical inference process go further. These alignments are selected and evaluated by some rough similarity measure, for which we chose the distributional similarity calculated from the Google n-gram corpus. A final score for each T/H pair is given by a classifier which uses the evaluation of alignments and their contribution to the inference process as features, together with shallow ones such as word overlap. The classifier is trained on the PASCAL RTE dataset.

With the *Text* obtained in the previous component, we generated four T/H pairs for each question. The *Hypothesis* can be extracted easily from the candidate answers. These pairs are inputs to our Recognizing Textual Entailment component. We selected the highest pair as the output for our system.

## 3 Results and Evaluations

### 3.1 Test Set and Evaluation Measure

The test data for evaluation in the Entrance Exams shared task were taken from the Japanese university entrance examination. Nine test documents were provided in the task. Eight contained five questions each, and the other contained six questions. Four candidate choices were shown for each question. The following is an example question with its four answer options.

- **Question.** *Where did the author's mother sit when one of her children was away?*
- **Answer 1.** *She didn't change her chair.*
- **Answer 2.** *She moved her own chair next to Dad's.*
- **Answer 3.** *She moved to an empty chair on the side.*

– **Answer 4.** *She sat opposite to Dad.*

The task evaluates each participating system by giving them a score between 0 and 1. The measure is called c@1 and was used in previous QA4MRE tasks. Systems might obtain higher scores if they leave questions unanswered when they may possibly be wrong. The measure is defined as follows.

$$C@1 = \frac{1}{n}(n_r + n_u \frac{n_r}{n}) \tag{1}$$

where:

– $n$: the total number of questions
– $n_r$: the number of correctly answered questions
– $n_u$: the number of unanswered questions

### 3.2 Results

We submitted five runs, NII1 to NII5, in the task. NII1, NII3 and NII5 answered all given questions, while NII2 and NII4 left several questions unanswered based on the score from the Recognizing Textual Entailment component. NII1 and NII2 generated the *Text* by using the *Sentence Extractor*, but NII3 and NII4 generated them just by using all the documents. NII5 combined results from NII1 and NII3. The evaluation results are listed in Table 2.

**Table 2.** System Results

| System | Correctly Answered | Incorrect Answered | Unanswered | c@1 |
|---|---|---|---|---|
| NII1 | 11 | 35 | 0 | 0.24 |
| NII2 | 7 | 17 | 22 | 0.22 |
| **NII3** | **16** | **30** | **0** | **0.35** |
| NII4 | 8 | 19 | 19 | 0.25 |
| NII5 | 15 | 31 | 0 | 0.33 |

From this table, NII3 had the highest score. It answered all 46 questions and 16 of them were answered correctly. Except for NII3 and NII5, other runs had scores similar to the random baseline, which is a 0.25 C@1 score. Besides the main C@1 score, we also applied McNemar's test to find whether the difference between baseline and NII3 or NII5 runs would be regarded as significant enough, but we found the p-value was still not sufficient in this case. We expected NII1 or NII5 to have the best scores before the evaluation since we expected the *Sentence Extractor* to remove noise in the documents and contribute to accuracy of the Recognizing Textual Entailment component. It turned out that the *Sentence Extractor* component became bottlenecked in NII1. Compared with NII1, NII3 acquired detailed information from the entire document, which resulted in a good c@1 score. We analyze error reasons for the *Sentence Extractor* in the next subsection.

### 3.3 Error Analysis

We carried out an error analysis for our system. We described in the previous section that our system architecture consists of three components, and each one possibly affected the final results. The *Character Resolver* performed well because most mentions for story characters seemed to have been tagged correctly. The main reason is that there were few story characters in the documents, so it was easy to tag them even just based on simple features. The major causes for errors in this task came from the *Sentence Extractor* component and Recognizing Textual Entailment component. Two kinds of errors need to be taken into account regarding the *Sentence Extractor*. The first type is that the component extracted wrong sentences because there was not enough information in the question. One example is shown as follows.

- **Question:** *What was the purpose of the wooden cat?*
- **Extracted sentences:** *When the touch was repeated a moment later, she whispered to her husband about it , but he only replied that people do not bother you at the opera on purpose. (and other four sentences)*
- **Correct sentences:** *Thieves had stolen the jewels and were going to pass them over to a woman. In order to identify her, they placed the cat in the booth and told her to pick it up.*

In the example, very few words are contained in the question. As stated in the previous section, our approach extracts sentences mainly depending on the similarity between sentences and questions, so we give the extracted sentences a very high score if they contain the same words appearing in the question, i.e., "purpose". In this case, the correct sentences also contains the same word as in the question, i.e., "cat". Since we give each word a tf-idf weight, a common word like cat contributes very little here. Half of the *Sentence Extractor* errors belong to this type. If we can deal with abstract words such as "purpose", we might obtain better results.

The other error type of the *Sentence Extractor* is the extracted sentences were not sufficient to answer the question, since we restricted the component to only extract five sentences. This error occurs when the number of relevant sentences is more than five sentences. An example of such cases is shown below.

- **Question:** *What happened when the author used a cash machine?*
- **Extracted sentences:** *When I first tried to use a cash machine in a bank, I had an unpleasant experience. (and four other sentences)*
- **Correct sentences:** *When I first tried to use a cash machine in a bank, I had an unpleasant experience. (and* **7** *other sentences)*

For this example, our system extracted some of the necessary sentences, but the questions asked about the information which corresponded to a larger range.

We failed to acquire all necessary information in similar cases, which might have reduced the accuracy for the Recognizing Textual Entailment component. These two reasons degraded the performance of our NII1 and NII2.

The error for the Recognizing Textual Entailment component is a common problem. Even in the recent workshop [9], a state-of-the-art system could not solve difficult questions. As described in the introduction section, the Entrance Exams task requires a high level of common sense, which high school students are expected to have, but our component has not reached such a level. The following is an example.

- **Text:** *Our plane would have to fly hundreds of miles out of our way to get around it. If we flew through the cloud, the engines might get full of ash and stop.*

- **Hypothesis:** *The pilot had to fly off the regular course for the sake of safety.*

A human would easily come to the conclusion that the hypothesis is correct, but when it comes to the Recognizing Textual Entailment component, it seems that it is difficult to solve this kind of problems. This also reduced the accuracy of our system.

## 4 Conclusion

We described our system that was used in the Entrance Exams task of QA4MRE CLEF 2013. Our system consists of three components, *Character Resolver*, *Sentence Extractor* and Recognizing Textual Entailment. In our developed system, the documents are processed by the *Character Resolver* to tag each story character an ID. The *Sentence Extractor* then extracts related sentences for each question and creates a *Hypothesis* and *Text*. Finally it inputs this T/H pair into the Recognizing Textual Entailment system to select an answer.

The best run of our system was NII3, which obtained 0.35 in c@1 score. This run used the entire document text as a *Text*, which helped to collect useful information for our Recognizing Textual Entailment component. The errors resulting from the *Sentence Extractor* and Recognizing Textual Entailment components negatively affected the accuracy of our system. Mitigating the limitations in these two components will be our future work.

## References

1. Peñas, A., Hovy, E., Forner, P., Rodrigo, Á., Sutcliffe, R., Forascu, C., & Sporleder, C. (2011). Overview of QA4MRE at CLEF 2011: Question answering for machine reading evaluation. Working Notes of CLEF, 2011.
2. Bhaskar, P., Pakray, P., Banerjee, S., Banerjee, S., Bandyopadhyay, S., & Gelbukh, A. (2012, September). Question answering system for qa4mre@ clef 2012. In CLEF (Online Working Notes/Labs/Workshop).

3. Iftene, A., Gînsca, A-L., Moruz, M.A., Trandabat, D., Husarciuc, M., Boros, E. "Enhancing a Question Answering System with Textual Entailment for Machine Reading Evaluation." CLEF (Online Working Notes/Labs/Workshop) 2012.
4. Shima, H., Kanayama, H., Lee, C. W., Lin, C. J., Mitamura, T., Miyao, Y., ... & Takeda, K. (2011, December). Overview of ntcir-9 rite: Recognizing inference in text. In Proceedings of the 9th NII Test Collection for Information Retrieval Workshop (NTCIR'11) (pp. 291-301).
5. Klein, Dan, and Christopher D. Manning. "Accurate unlexicalized parsing." Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1. Association for Computational Linguistics, 2003.
6. Finkel, Jenny Rose, Trond Grenager, and Christopher Manning. "Incorporating non-local information into information extraction systems by gibbs sampling." Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2005.
7. Lee, H., Chang, A., Peirsman, Y., Chambers, N., Surdeanu, M., & Jurafsky, D. (2013). Deterministic coreference resolution based on entity-centric, precision-ranked rules. Computational Linguistics, (Just Accepted), 1-54.
8. De Marneffe, Marie-Catherine, Bill MacCartney, and Christopher D. Manning. "Generating typed dependency parses from phrase structure parses." Proceedings of LREC. Vol. 6. 2006.
9. Dzikovska, M. O., Nielsen, R. D., Brew, C., Leacock, C., Giampiccolo, D., Bentivogli, L., ... & Dang, H. T. (2013, June). SemEval-2013 task 7: The joint student response analysis and 8th recognizing textual entailment challenge. In Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013), in conjunction with the Second Joint Conference on Lexical and Computational Semantcis (* SEM 2013), Atlanta, Georgia, USA, June. Association for Computational Linguistics.