sJanta: An Open Domain Question Answering System

Md. Arafat Rahman[1] and Md-Mizanur Rahoman[1,2]

[1] Department of Computer Science & Engineering,
Begum Rokeya University, Rangpur, Bangladesh.
arafatrahmanbrur@gmail.com
[2] Department of Informatics,
The Graduate University for Advanced Studies, Tokyo, Japan.
mizan@nii.ac.jp

Abstract. This paper reports on the participation of the system namely "sJanta" at NII Testbeds and Community for Information Access Research Project 11 (NTCIR-11) Question Answering (QA)-Lab English sub-task. sJanta is a modular question answering system that can answer multiple choice questions given in English natural language. We use English Wikipedia as knowledge-base. At first we retrieved Wikipedia articles for the question. After finding the Wikipedia articles, we retrieved question specific passages. Then, we did dependency parsing, question context analysis and semantic similarity matching to score the answer choices. Finally, the best score-generating answer choice was picked as the answer. Although sJanta applied very simple technique in answering the questions, performance of the system is quite promising.

Keywords: #NTCIR11, QA-Lab, Question Answering System

1 Introduction

Currently large amount of textual data are kept on a variety of digital mediums such as digital archives, the Web or the hard drives of our personal computers which hold huge knowledge-base. However, because of availability of sheer amount of textual data, traditional document-based information access (IA) sometime is not sufficient [1]. This is because, over traditional document-based IA, users receive related documents rather than the required information which are still quite large. Therefore, specific IA (such as Question-Answering (QA)-based IA) is more pragmatic choice and was investigated in several contemporary researches such as [2–4]. However, most of such works do not take care of contextual information attachment. But in a real-world scenario, context plays a great role in IA. Therefore, over QA-based IA works, context-awareness is a desired attribute.

To investigate context-aware QA, NTCIR11 introduced QA-Lab sub-task. In this sub-task, participants needed to answer questions for a given context. Organizer used Japanese University Entrance Exam History course questions

as question set. In answering the questions, sub-task organizer did not restrict to use a particular knowledge-base. Therefore, in this sub-task, knowledge-base could be used is quite large. Considering the requirement of the task, we realized that sub-task organizer tried to address the insufficiency of document-based IA. Moreover, as Japanese University Entrance Exam question answering requires context-awareness, organizer also interested in QA systems which able to embed question context. In this work, we addressed the both issues and reported our participation of NTCIR11 QA-Lab English sub-task. With preliminary implementation of our system (i.e., sJanta), we participated QA-Lab pilot sub-task for multiple choice questions. We found that although our proposed system is simple, but it is promising.

In sJanta, we used linguistic-based approach to understand the questions. As knowledge-base, we use on-line Wikipedia. For each question, we did wikification on the question and the context of the question. Wikification is a technique which tags Wikipedia articles for text. After finding the Wikipage articles, we retrieved question specific passages. Then, we did dependency parsing, context analysis and semantic similarity matching to score the answer choices. Finally, the best score-generating answer choice was picked as the answer.

The best features of sJanta - i) it adapts context awareness of question ii) it leverages existing NLP tools iii) it works on fully unsupervised-basis iv) it is simple.

The remainder of this paper is divided as follows. Section 2 describes proposed system in details. In section 3 we show the results of implementing proposal through experimental results and discussion. Finally, section 4 concludes our study.

2 System Architecture

The overall architecture of sJanta is shown in figure 1 and consists in two major components: a “Question Analysis Module” and a “Answer Generator Module”.

The Question Analysis module parses question instructions, context, and answer choices from question file and generates hypotheses (will be described later), Named Entities (NEs) and Wikipage links. The NEs and Wikipage links are used in the Answer Generator module to retrieve Wikipedia articles that talk about the hypotheses.

On the other hand,the Answer Generator module collects Wikipedia articles using NEs and Wikipage links. Then, it generates score for each hypothesis based on retrieved Wikipedia articles. Finally, it makes a ranking of hypotheses based on their score thus selects the best ranked hypothesis as answer.

2.1 Question Analysis Module

The question analysis module has two sub-modules: a “Question Parser” and a “Wikifier”.

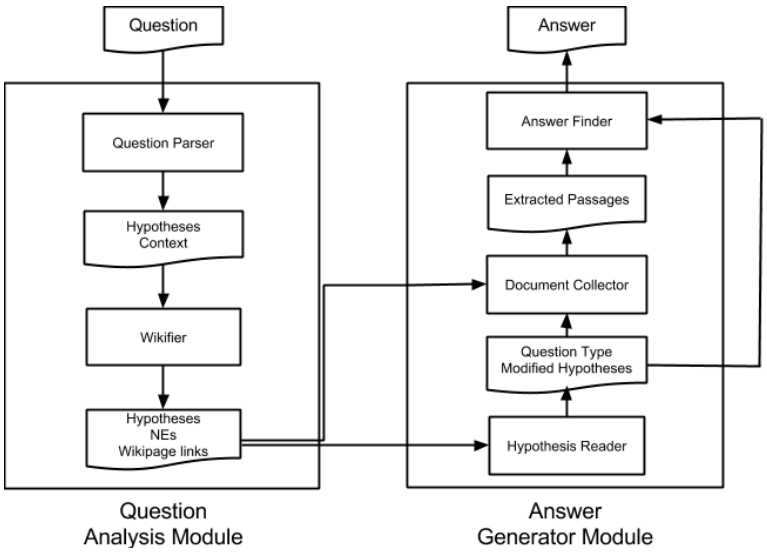

Fig. 1. Architecture of sJanta

Question Parser The Question Parser sub-module parses information including question instruction, context and answer choices from question file. Then, it generates hypotheses from parsed information and sends them to wikifier sub-module. Hypotheses are context-aware answer choices. We generated hypotheses by adding context towards the answer choice. We add context at the first of the each answer choice. QA-Lab sub-task organizer provided question context with some tags. For example, if given question context is "Christianity" and a answer choice is "It was established based on the teachings of Jesus, who criticized Islam", we created hypotheses as "Christianity was established based on the teachings of Jesus, who criticized Islam".

Wikifier The Wikifier sub-module runs wikification on the hypotheses and contexts and generates a wikified xml file as its output. The wikified xml file contains question instructions, hypotheses, NEs and wikipage links. Later, this wikified xml file is used by Answer Generator module to generate answers.

2.2 Answer Generator Module

The Answer Generator module has three sub-modules: a "hypothesis reader", a "document collector", and a "answer finder" sub-module.

Hypotheses Reader The hypothesis reader sub-module parses the wikified xml file (i.e., the output of Wikification) and then, detects the question type:

Affirmation or Negation. Affirmation means question asking about positive answer while Negation means question asking about negative answer. For example, "Which country did not take part in World War II?" is a Negation type question.

Document Collector The document collector sub-module extracts Wikipedia articles that talk about the hypotheses. Wikitext extractor (e.g., jsoup[3]) is used to find Wikipedia articles. The retrieved Wikipedia articles are then sent to answer finder sub-module to generate answer.

Answer Finder The answer finder sub-module extracts nouns and verbs from both the hypotheses and the retrieved passages. After extracting nouns and verbs, it generates scores namely "noun score" and "verb score" based on checking Word-Net based similarity between the nouns and verbs of the hypotheses and the retrieved paragraphs. Then, it does a dependency parsing for both hypotheses and retrieved paragraphs to extract relationships of nouns and verbs with other words in the sentences. Then, it generates a dependency score based on the dependency relationships of words in the sentences of both the hypotheses and the retrieved text. Later, it combines noun score, verb scores and dependency score to generate a final score for each hypotheses. Then, it makes a ranking of hypotheses based on their final scores. Finally, it selects a hypotheses as answer based on their rankings and question type i.e Affirmation or Negation. If the question type is Affirmation then, it takes the highest ranked hypotheses as answer otherwise the lowest ranked one as answer.

3 Experiment

In the experiment, we used NTCIR11 QA-lab English sub-task questions. NTCIR11 provides Japanese University Entrance Exam History course questions for the year 2003 and 2007. Each question holds a context, question description and 4 answer choices in natural language text. Context of the question is given with tag called "Underline Text". However, we excluded some questions. They are: i.) questions which had multiple sentences answer choice ii) questions which required to select images etc. We consider them as "OUT OF SCOPE" of sJanta.

Below listing shows an exemplary question. Here first part of the listing is shown for context (i.e., tagged by <uText>) while second part of the listing is shown for question (i.e., tagged by <instruction>) and answer choices (i.e., tagged by <choice>).

```
... an important place of pilgrimage in
        <uText id="U1"><label>(1)</label>
                Christianity
        </uText>.
In the 9th century ...
```

[3] http://jsoup.org/

```
...
<instruction>
        From 1-4 below, choose the one sentence that
        correctly describes the underlined portion
        <ref comment="" target="U1">(1)</ref>.
</instruction>
...
<choice ansnum="1">
        <cNum>(1)</cNum>It was established based
         on the teachings of Jesus, who criticized Islam.
</choice>

<choice ansnum="2">
        <cNum>(2)</cNum>Its holy book is the New Testament.
</choice>

<choice ansnum="3">
        <cNum>(3)</cNum>It was made the state religion by
        the Emperor Diocletian.
</choice>

<choice ansnum="4">
        <cNum>(4)</cNum>Wycliffe was declared a heretic at
         the Council of Clermont (ecumenical council).
</choice>
...
```

We used on-line Wikipedia as knowledge-base. To get Wikipage links of Wikipedia, we use Illinois Wikifier[4]. To get Wikipedia articles from Wikipage links, we used Wikitext extractor called jsoup[5]. To parse text of Wikipage links, we use stanford dependency parser[6]. To calculate semantic similarity of word of the text, we use Word-Net Similarity for Java (ws4j)[7].

Below we report MCQ (multiple choice question) results for Phase1 and Phase2.

3.1 Phase 1 MCQ

In the phase 1, NTCIR11 QA-Lab English sub-task provided 35 questions. Among them, we answered 26 questions and excluded 9 questions (i.e., A8, A10, A15, A18, A19, A20, A22, A24, A29). For every question, NTCIR organizer

[4] http://cogcomp.cs.illinois.edu/page/software_view/Wikifier
[5] http://jsoup.org/
[6] http://nlp.stanford.edu/software/stanford-dependencies.shtml
[7] https://code.google.com/p/ws4j/

Table 1. Performance of sJanta over NTCIR QA-lab Phase 1 question set

Total Qs	# of Qs Answered	# of Correct Answer	% of Correct Answer w.r.t. Answered Qs	Score w.r.t. Total Qs (i.e., NTCIR Score)	Score w.r.t. Answered Qs
35	26	13	50.00	36 out of 97	36 out of 71

Table 2. Performance of sJanta over NTCIR QA-lab Phase 2 question set

Total Qs	# of Qs Answered	# of Correct Answer	% of Correct Answer w.r.t. Answered Qs	Score w.r.t. Total Qs (i.e., NTCIR Score)	Score w.r.t. Answered Qs
41	30	10	33.33	23 out of 100	23 out of 73

fixed a score either 3 or 2. In our understanding, score signifies difficulty level of question. Summation of score for these 35 questions were 97 and summation of our answered questions were 71.

Table 1 shows the performance. Among the answered questions (i.e., 26), we got correct result for half of them (i.e., 13). Considering all questions (whether we answered or not), our correct answer scored 36 point out of 97 points. However, if we consider the answered question only, it was 36 out of 71. We conclude, for half of the questions, we could understand the correct context.

3.2 Phase 2 MCQ

In the phase 2, NTCIR11 QA-Lab English sub-task provided 41 questions. We answered these 30 questions and excluded 11 questions (i.e., A5, A11, A12, A17, A18, A19, A24, A30, A34, A37, A40). Summation of score for these 41 questions were 100 while summation of score that we answered was 73.

Table 2 shows the performance. In phase 2, we performed poorly. Among the answered questions (i.e., 30), we got correct result for only one third (i.e., 10). Considering all questions (whether we answered or not), our correct answer scored 23 point out of 100 points and considering the answered question only, it was 23 out of 73. Here we faced problem of picking correct context which ultimately leads poor performance.

4 Conclusion

In this paper, we discussed an open domain question answering system namely sJanta as a participant of NTCIR-11 Japanese Entrance Exam task. In this task, our system is asked to read a passage that talks about a specific topic and instructed to answer a set of questions. The questions are given in multiple choice format, with multiple choices from which one choice need to be selected as answer. The questions are highly contextually aware and are more suited to human beings rather than a computer system. So, the task was really a

challenging problem to investigate. To solve the problem we used a linguistic based approach to understand the questions and extracting answers for it. We used currently existing NLP tools which showed scalability option of our system - therefore it is simple as well.

There are a lot of issues that could improved in our system as future works. The multiples hypotheses may be generated for every answer choice. We may also generate scores both for rejecting an answer choice and accepting the answer choice, and then, combine both the scores to get a final score. Apart from MCQ, in future, we want to investigate our system to extend for other types of questions.

References

1. J. Allan, B. Croft, A. Moffat, and M. Sanderson. Frontiers, challenges, and opportunities for information retrieval: Report from swirl 2012 the second strategic workshop on information retrieval in lorne. *SIGIR Forum*, 46(1):2–32, May 2012.
2. D. A. Ferrucci. Introduction to "this is watson". *IBM J. Res. Dev.*, 56(3):235–249, May 2012.
3. M. Iyyer, J. Boyd-Graber, L. Claudino, R. Socher, and H. Daumé III. A neural network for factoid question answering over paragraphs. In *Empirical Methods in Natural Language Processing*, 2014.
4. M.-M. Rahoman and R. Ichise. Automatic inclusion of semantics over keyword-based linked data retrieval. *IEICE Transactions of Information and Systems*, E97-D(11):2852–2862, 2014.