Multi-choice Question Answering System of WIP at NTCIR-12 QA Lab-2

Bingfeng Luo
ICST, Peking University
bf_luo@pku.edu.cn

Yuxuan Lai
ICST, Peking University
erutan@pku.edu.cn

Lili Yao
ICST, Peking University
yaolili@pku.edu.cn

Yansong Feng
ICST, Peking University
fengyansong@pku.edu.cn

Dongyan Zhao
ICST, Peking University
zhaody@pku.edu.cn

ABSTRACT

This paper describes a multi-choice question answering system we designed for the NTCIR-12 QA Lab[3]. This system aims at analysing and answering world history multi-choice questions in the Japanese National Center Test (in English). Our system utilizes preliminary results from an information retrieval baseline as a starting point, and improves by taking structured knowledge base as well as additional time constraints into consideration. In the final evaluation, we achieved 34 points on the 2011 test dataset.

Team Name

WIP

Subtasks

National Center Tests, Formal Run (English)

Keywords

Question Answering, Multi-Choice Question, Knowledge Base, Information Retrieval

1. INTRODUCTION

The QALab of NTCIR-12 requires participant systems to answer world history questions in the Japanese university entrance exams. This task includes two separate datasets: National Center Test and Second-stage Examinations, each of which has both Japanese version and English version. The former dataset only consists of multi-choice questions, and the second one, which is usually required by some Japanese universities after the National Center Test, consists of various types of questions, including multi-choice, slot-filling, essay and etc. In this paper, we focus on the English version of the National Center Test dataset.

There are in total 4 types of multi-choice questions in the National Center Test dataset, including factoid questions, slot-filling questions, True-or-false questions and unique questions. Factoid questions require to choose correct entities or events with regard to the instruction. Slot-filling questions require a system to fill in the blanks in a given context, which is usually several paragraphs of words providing background information about a historical entity, event or a certain period. True-or-false questions will provide several statements about a given topic, and a system is expected to choose correct or wrong statements. Unique questions are the ones that can not be included in previous 3 types, and usually have various forms. For example, a system may need to choose a sequence of events which occur in a chronological order, or a system is required to answer questions according to a given picture.

The questions in this task are not self-contained. The context information provided to each question is not enough to induce the right answer and participant systems are allowed to utilize whatever external knowledge sources to facilitate their question answering process. The organizer of this task also provide some useful knowledge sources: Wikipedia world history articles, annotated Japanese high school textbooks, and a world history ontology. However, the textbooks are only provided in Japanese and only a fraction of the ontology is translated into English (semi-automated). So we can only use the Wikipedia articles and the translated fraction of the world history ontology in our system.

The rest of this paper is organized into 3 sections. We describe our system architecture in Section 2. We show some experimental results of our system in Section 3. Finally, we draw a conclusion in Section 4.

2. SYSTEM ARCHITECTURE

2.1 Overview

We consider multi-choice question answering as a multi-true-or-false problem. To be specific, we first preprocess the data and convert each option of a question into one or two assertions by combining the option, question instruction and the corresponding context. Then we use our information retrieval module to give each assertion a confidence score indicating to what extent this assertion is true. After that, we use knowledge base information to rerank the assertions. The top ranked assertion is our finally choice. Additionally, we treat chronological sequence questions as a special case by going through a time constraint module.

2.2 Preprocessing

The preprocessing module aims at converting each option of a question into its corresponding assertion. To achieve this, we first use hand-crafted rules to classify each question into several predefined types. Then, we apply different rules/patterns for each question type to generate the result assertion. Finally, we induce the polarity of each question, indicating whether we should pick the most plausible or most implausible assertion as our final choice.

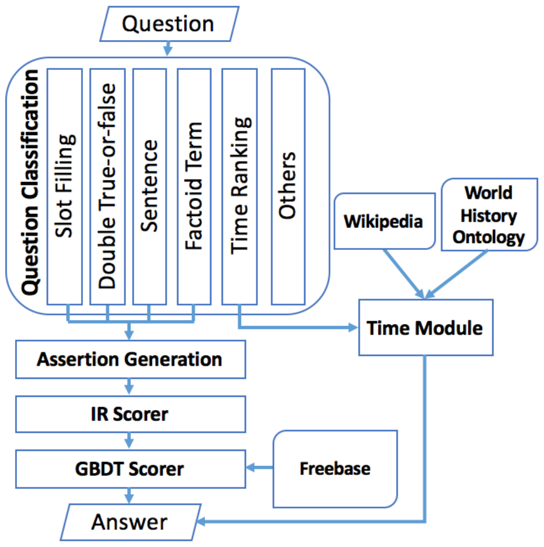

Figure 1: System Overview

2.2.1 *Question Type Classification*

In this evaluation, we consider 5 question types, and ignore other types (especially, those highly relying on figure/image analysis). Since the data is well structured in XML format, we can therefore utilize structure information to determine question type.

- Slot-filling: Almost all slot-filling questions will mention the blanks it requires to fill in the question instruction. So we can simply detect whether there exists at least one blank reference in the question instruction to decide whether it belongs to this type.
- Double True-or-false (two sentences): This is different from vanilla True-or-false questions in that it displays two sentences in the question instruction, and asks a system to judge the correctness of both of these sentences. The options are "Correct, Correct", "Correct, Incorrect" and so on. This kind of questions has an question data section which contains the two assertions in a form of list. Therefore, if there are exactly two items in the question data list, then we will classify it as Double True-or-false question.
- Chronological Sequence: In this type, a system is asked to arrange several historical events in a chronological order (starting from the oldest one). This kind of questions often display all events in the question data section in a form of list. Therefore, we classify questions whose question data section has a list containing more than 2 items as this type. Additionally, we also detect " - " pattern (space-hyphen-space, used as a separator of entries to be ranked) in the option. If there are at least two of this patterns in each option, then we also classify this question as Chronological Sequence type.
- True-or-false: In this type, options are in a form of assertions and a system is required to pick out the correct or incorrect one. We classify a question into this type if it does not belong to the previous 3 types, and it has at least 5 words. The 5-word limitation is chosen empirically and works well in practice.
- Factoid Term: In this type, options are entities or historical events. We need to find the correct one with respect to the question instruction. If a sentence doesn't belong to any of the 4 types above, then it is classified into this type.

2.2.2 *Assertion Generation*

The assertion generation process is purely rule-based. Note that we don't generate assertions for chronological sequence questions, which will be treated specially in Section 2.5.

- Slot-filling: We fill the blanks with the option, and then choose the sentence that contains the filled blanks as assertion.
- Double True-or-false (two sentences): We generate two assertions for this type, each with one of the two sentences.
- True-or-false: We simply use the raw sentence as the assertion.
- Factoid Term: If the question instruction refers to an underlined sentence in the context, then we concatenate the underlined sentence, the instruction, and the option to form the assertion. Otherwise, we only concatenate the instruction and the option to form the assertion.

Note that the generated assertion may not be a valid sentence. Therefore, syntactic analysis may not be suitable for them.

2.2.3 *Question Polarity*

Some questions ask a system to pick the correct sentence from all options, while the others ask a system to pick the incorrect one. Therefore, to make the final option choice, we need to get the polarity of each question.

Following [4], we also use a rule-based method to decide the polarity. We detect the following phrases in the question instruction: *that incorrectly, that was not, incorrect sentence, wrong sentence, is incorrect, is wrong, incorrectly describes, contains a mistake*. If at least one of these phrases is found, then we will consider the polarity of this question as negative. In other words, for these negative questions, we will pick the assertion with lowest score as our final choice.

2.3 Information Retrieval Method

The basic score of each assertion is obtained by an information retrieval method. Instead of using the world history part of Wikipedia articles only, we utilize the full Wikipedia dump to obtain more comprehensive information. We apply Lucene to index each sentence in the dump and use a standard analyzer to generate tokens to be indexed. We use the assertion as query, and collect the highest score of returned sentence as the score of each assertion.

2.4 Knowledge Base Method

We use Freebase as our structured knowledge base, and Illinois Wikifier[2] to identify entities in the assertion. Since Illinois Wikifier gives Wikipedia ID as result, we use $\backslash wikipedia\backslash en\_id$ predicate in Freebase to match Wikipedia ID to Freebase

Feature	Description
bi-score	number of Freebase connections between entities of question side and answer side, divided by the minimum number of entities of either side
assertion-score	number of Freebase connections among entities within assertion, divided by the number entities in the assertion
question-entity	number of entities in the question instruction
option-entity	number of entities in the option
assertion-entity	number of entities in the assertion
option-len	number of words in the option
assertion-len	number of words in the assertion
is-sentence	the option is a sentence or not
is-slot	the question is a slot filling question or not
is-term	the question is a factoid-term question or not

Table 1: Features used in GBDT. Question side entities include entities in the question instruction, and entities of the underlined sentence that this question refers. Answer side entities are those entities in the assertion.

mid. And we also filter out entity mentions that are contained by another entity to reduce noises. After that, we draw two statistical features with respect to the connection between these entities in Freebase (see the first two features in Table 1).

To further calibrate the score generated by information-retrieval method, we employ the learning-to-rank paradigm. To be specific, given a question with k options, we collect all pairs of options as training data. In each pair, if the first option is true and the second option is false, then its class is *true* (the first option is better than the second one). Otherwise, its class is *false*. We use gradient boosting decision tree (GBDT) as the classifier, and the features are shown in Table 1. Note that all the features are collected for both options, and an additional set of features are generated by doing subtraction operation between the feature sets of the first option and the second option (except for the last 3 features). During prediction phase, for each option, we count how many options are predicted worse than the given option and use the count as its finally score.

2.5 Time Constraint

Time information is used to solve chronological sequence questions, which usually first list some historical events, and then ask a system to rank them in a chronological order.

We first collect all the historical events and entities that have corresponding time span information in the English version of the Word-History Ontology (only a fraction of full Japanese version). Then we perform string matching of these events and entities using Illinois Wikifier to detect historical event or entity mentions. For each event or entity, we use their starting time to rank them. For those entities or events that do not have time information in the ontology, we go the their Wikipedia page and use hand-crafted rules to extract their starting time from the Wikipedia's infobox. Additionally, we also use regular expressions to detect time mentions in colons of each entry, and simply use this as its time if detected. If an entry has more than one detected entities or events, we simply use the first entity to rank.

Method	Precision
Gradient Boost Decision Tree (GBDT)	51.0
Pure Information Retrieval (IR)	49.5

Table 2: Comparison of GBDT and IR method.

3. EXPERIMENTS

3.1 Experimental Setup

We use 1997, 2001, 2003, 2005, 2007, 2009 National Center Test dataset for training, and abandon those questions highly rely on figure/image processing. During training, we use cross validation and grid search to select parameters. And cross validation is performed by taking each year for validation one by one.

We use the sklearn[1] implementation of GBDT. As for parameters, we use 10 estimators, learning rate of 0.1, maximum depth of 3 for individual regression estimators.

3.2 Main Results

To show how our GBDT method improves information retrieval scores, we compare the cross validation result of a GBDT method with the result of our pure information-retrieval method. As seen in Table 2, the GBDT method improves the overall precision by 1.5 points.

As for chronological sequence questions, there are actually only 3 such questions in the training data. Therefore, we manually analyzed the performance of our method on these three questions. Our method performs well on *1997 Q10* and *2007 Q33* but fails on *2003 Q41*. The time information mainly comes from Wikipedia in *1997 Q10*, and mainly comes from ontology in *2007 Q33*. Since we are unable to detect enough useful entity or event mentions in *2003 Q41*, our method failed on that question.

In the test phase of NTCIR-12 QALab (phase 3), the 2011 National Center Test is used. Our system achieved 34 points (precision of 33.3%) and ranked the 3rd place. Among the 3 chronological sequence question/sentence , our system correctly solved one of them. The failure on the other two questions are caused by the reason that we can't detect useful entity or event mentions for some entries, and some detected mentions don't have time information in Wikipedia or ontology.

4. CONCLUSION

We described our system at the NTCIR-12 QA-Lab Task in this paper. Our system use GBDT classifier to combine the result of information retrieval method and the information of knowledge base. We also utilize time information of entities and events from world history ontology and Wikipedia to solve chronological sequence questions. The system achieved 34 points (precision of 33.3%) in the 2011 Center Test tasks, and our team ranked the 3rd place in the final evaluation.

To get further improvement, we plan to explore better methods to utilize structured knowledge base, build stronger entity and event linking tool, and find more resources to supply time information for historical events and entities.

5. REFERENCES

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[2] L. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1375–1384. Association for Computational Linguistics, 2011.

[3] H. Shibuki, K. Sakamoto, M. Ishioroshi, A. Fujita, Y. Kano, T. Mitamura, T. Mori, and N. Kando. Overview of the ntcir-12 qa lab-2 task. In *Proceedings of the NTCIR-12*. NII, 2016.

[4] D. Wang, L. Boytsov, J. Araki, A. Patel, J. Gee, Z. Liu, E. Nyberg, and T. Mitamura. Cmu multiple-choice question answering system at ntcir-11 qa-lab. In *NTCIR*, 2014.