TREC 2003 Question Answering Track at CAS-ICT

Yi Chang, Hongbo Xu, Shuo Bai
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
changyi@software.ict.ac.cn
http://www.ict.ac.cn/

Abstract

In our system, we make use of Chunk information to analyze the question. A multilevel method is fulfilled to retrieve a candidate Bi-sentence. As to answer selecting, we proposed a voting method. We figure out the performance of each module, and our study shows that 65.54% information has lost in document retrieval and Bi-sentence retrieval.

Keyword: Chunk, Multilevel retrieval, voting

1. Introduction

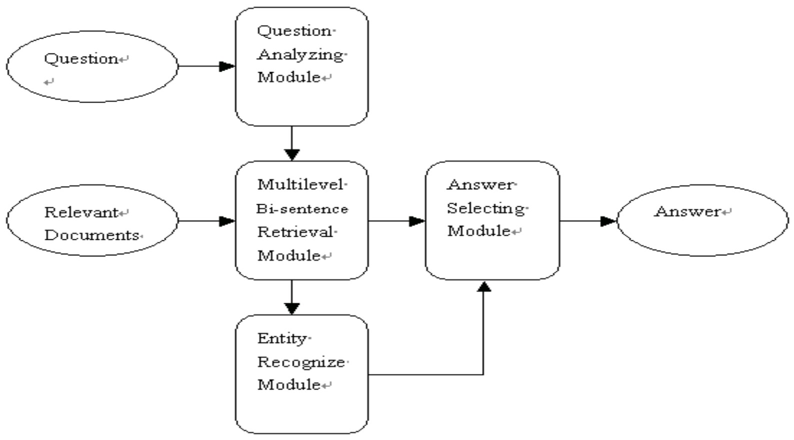
It is the third time we take part in the TREC-QA track. We undertake the main subtask and submit three runs for evaluation.

Our QA system incorporates several useful tools. The first is LT_CHUNK that is developed at University of Edinburgh. We use LT_CHUNK to get the chunks of the sentence and the POS tags of different words. The second is GATE that is developed by University of Sheffield. We use GATE to identify some Named Entities.

In our previous QA system, we tried different kinds of elaborate algorithms, but the results were not satisfactory, and we didn't make it clear what the performance of each step in our system is. So we try to figure them out this year, and our study shows: 65.54% information lost in document retrieval and Bi-sentence retrieval.

2. System Description

Our system contains four major modules, namely Question Analyzing Module, Multilevel Bi-sentence Retrieval Module, Entity Recognizing Module and Answer Selecting Module. However, for those definition questions, the Entity Recognizing Module is unnecessary. The system architecture is represented as below.



To answer each question, Question Analyzing Module makes use of NLP technique to identify the right type of information that the question requires. We select top 50 out of the 1000 given relevant documents to find the candidate answer from them. Since there exists too much redundant information in these documents, Multilevel Bi-sentence Retrieval Module matches the question with the 50 relevant documents at different levels and retrieves some top rank Bi-sentences to find the candidate answer from them. Entity Recognizing Module identities the candidate entities from the selected Bi-sentences, and Answering Selecting Module selects the answer in a voting method.

2.1 Question Analyzing Module

Since most of the factoid and list questions ask for Named Entity (NE) as the response, Question Analyzing Module tries to identify the required NE type while parsing a question.

Some question words can map to the required NE types directly, e.g. “who”(PERSON), ”where”(LOCATION), “how many”(NUMBER).

However, for most of the questions whose question word is “which” or “what”, we need to find the Core Noun to help us to identify the required NE type. Core Noun is a noun in each question that indicates the answer. For example:

[1] Which city is home to Superman?

[2] Which past and present NFL players have the last name of Johnson?

[3] What type of bee drills holes in wood?

The Core Noun of question [1] is “city” since the answer to the question is a city, and the Core Noun of question [2] and [3] is respectively “players” and “bee”. We identify the required NE type according to a predefined Map Lexicon that consists of hundreds of nouns mapping to the NE type that can be recognized later, e.g. “city”(CITY), “player”(PERSON). We build another Abstract Noun Lexicon that consists of some abstract nouns, e.g. “type”, “breed”.

The algorithm to find the Core Noun is as follows:

Step 1: Take the last noun in the first Noun Group as Core Noun;

Step 2: If the Core Noun is in Abstract Noun Lexicon, find the last noun in the next Noun Group as Core Noun;

Step 3: If there is no suitable noun that can be found, the Core Noun is empty.

However, there are some questions whose required NE type cannot be recognized later by Entity Identifying Module, e.g. “bee”. We regard these questions as “other NE questions” and keep their Core Noun for further matching.

The question whose Core Noun is empty is called “miscellaneous question”. For example:

[4] How did Minnesota get its name?

[5] What is tequila made from?

To use some syntactic patterns to find its answer is a practical method, but in our current QA system we simply answer NIL.

Processing the definition question is rather simple, and we just extract the phrase to be explained.

2.2 Multilevel Bi-sentence Retrieval Module

A Bi-sentence is a pair of consecutive sentences, and a Phrase is a sequence of keywords or one keyword in a question, where a Keyword is a word in the question but not in our Stop Word list.

The goal of Multilevel Bi-sentence Retrieval Module is to find some top rank Bi-sentences most relevant to a question. According to our training results on TREC-11 QA corpus, we select top 20 Bi-sentences for the factoid and definition question and top 50 Bi-sentences for the list question.

We assume: 1) Bi-sentences that can match a phrase of a question are more relevant than those only can match separate keywords. 2) Bi-sentences that can match a phase of a question in raw form are more relevant than those only can match in stemmed form.

Since the strict matching will decrease the recall rate of the Bi-sentences retrieval, we parse a Bi-sentence for several times and match the question at different levels to improve the precision rate without decreasing the recall rate.

For factoid and list question, our system applies a four-level method to select a candidate Bi-sentences. At each level, we define two kinds of substrings, Compulsory Phrase and Assistant Keyword. Compulsory Phrase is a phrase set in which each element is obligatory to match a Bi-sentences. Assistant Keyword is a keyword set in which each element is optional to match. Those words not belong to the Compulsory Phrase and Stop Word list are regarded as the elements of the Assistant Keyword. We compute the weight of a Bi-sentence as below:

$$weight\_p = \beta \times count\_a / (count\_q + count\_p)$$

where $weight\_p$ means the weight of the Bi-sentence, $count\_a$ means the number of matching Assistant Keyword between the question and the Bi-sentence, $count\_q$ means the number of keywords in the question, $count\_p$ means the number of keywords in the Bi-sentence and β is an experiential parameter. All relevant Bi-sentences are ranked: the Bi-sentence selected from the higher level has a higher priority, and in the same level, the Bi-sentence with a larger weight has a higher priority. Furthermore, the first level is based on raw matching, while the other three levels are based on stemmed matching.

At the first level, we take the last Noun Group and the last verb in the last Verb Group as the Compulsory Phrase. And those phrases with initial capital on each word are also regarded as the Compulsory Phrase. At the second level, we move the verb from the Compulsory Phrase to the Assistant Keyword because the verb is difficult to match and we don’t fulfill the verb expansion. At the third level, we only leave those phrases composed of successive initial capital words as the Compulsory Phrase. At the last level, the Compulsory Phrase is empty, all words belong to Assistant Keywords, and this is equal to the method in our previous QA system to compute relevance between the question and the candidate Bi-sentences.

For definition question, our system simply takes a two-level method to select a candidate Bi-sentence. At the first level, we take the phrase to be explained as the Compulsory Phrase. A Bi-sentence selected not only matches the Compulsory Phrase but also matches the definition pattern proposed by InsightSoft[1]. In the second level, we just regard each keyword in the question as Assistant Keyword, and then find the most relevant Bi-sentences according to the weight.

2.3 Entity Recognizing Module

We use GATE to recognize some types of Named Entities, such as PERSON, LOCATION, COUNTRY, etc. And we take some new strategies to recognize other simple types, such as YEAR, COLOR, DISEASE, etc.

For those questions whose required NE type is identified, we recognize the required NE as our candidate answer. However for those questions whose required NE type cannot be identified, we make use of Core Noun to construct the possible phrase as the candidate entity by some syntactic rules.

2.4 Answer Selecting Module

In the top 20 Bi-sentences of each factoid question, we may find more than one suitable Named Entity. How to choose the most suitable NE as the answer in our system is difficult, since the NE type is the only semantic information we used.

We assume that the answer in the top 20 Bi-sentences is most likely to appear for several times, so we use a voting method to select the answer. In our experiments on TREC-11 QA corpus, the voting method has an improvement of 15.58% comparing with the method that simply select the first one as the answer.

We also study the weighted voting method, where the weight of the candidate answer decreases with the rank of the Bi-sentence. However, in our experiments on TREC-11 QA corpus, the results are similar to the voting method above.

For list question, since the answer number of each question is not known, we choose the entities whose frequency in voting are beyond a threshold. According to training results on TREC-11 QA corpus, our threshold varies from the required NE type.

For definition question, we simply choose the first Bi-sentence as the answer.

3. Result

We don't try radically different methods in our three runs, and instead we simply make little change to get a steady output. Table 3.1 shows our results. In run A, we answer NIL for those "other NE question" whose required NE type is highly depended on Core Noun, and we only take part of the definition patterns into effect. In run B, we use some looser rules to construct the possible NE for "other NE question", while in run C we use some stricter rules.

RunID	Factoid Accuracy	#Correct (#NIL)	#Inexact	#Unsupported	List Ave_F	Definition Ave_F	Final Score
ICTQA2003A	0.128	53(21)	6	6	0.091	0.142	0.122
ICTQA2003B	0.140	58(17)	5	8	0.089	0.149	0.130
ICTQA2003C	0.145	60(14)	8	10	0.091	0.149	0.133

Table 3.1

4. Error Analysis

Here we just focus our analysis on factoid questions because it is easier to get a quantitative evaluation at each step than that of the other two types of questions. Since we simply answer NIL whenever we cannot find an answer, we have no opinion to the NIL questions. So we only focus our analysis on the question whose answer is not NIL. All of the error analysis is studied from the result of run C.

4.1 Question Analyzing Error

Table 4.1 shows the distribution of the question analysis and the error rate in each question category.

Question type	#Total Question	#NE identified Question/#Wrong	#Other NE Question/#Wrong	#Miscellaneous Question/#Wrong
Factoid	413	275/5	123/24	15/1
List	37	24/0	13/0	0/0
Definition	50	/	/	50/0

Table 4.1

The performance of Question Analyzing Module is fairly satisfactory, and the overall accuracy of question analysis is 94%. There are two main reasons for the rest 30 questions incorrectly analyzed: 1) 16.7% (5/30) error is caused by chunk error of LT_CHUNK. 2) 83.3% (25/30) error is that our Question Analyzing algorithm cannot cover all questions.

4.1 Retrieval Error

4.1.1 Document Retrieval Error

According to the our statistical result, there are 275 out of 413 factoid questions whose answers can be got from the top 50 documents. Since there are 383 questions whose answer is not NIL, the maximum accuracy of document retrieval with top 50 documents is 71.80%. In another word, there are at least 28.20% of 383 questions cannot be answered correctly in document retrieval process.

4.1.2 Bi-sentence Retrieval Error

According to our statistical result, from the top 20 Bi-sentences, there are only 132 out of 275 factoid questions can be answered correctly based on the answer and the correspondent Document ID. So the maximum accuracy of Bi-sentence retrieval by our Multilevel Bi-sentence Retrieval Module is 48.0%. Also, there are at least 52.0% of 275 questions cannot be answered correctly in Bi-sentence retrieval process.

4.2 Answer Error

There are the two main reasons for the inexact answer error: 1) 50% (4/8) error is caused by the inexact identification of required NE type; 2) 50% (4/8) is caused by the inexact recognizing of NE.

Since we don't make use of more semantic information than NE type, we cannot avoid the unsupported answer error.

According to our manual statistical result listed in Table 4.2, those questions whose required NE type can be identified are easier to answer.

Question Type	#Not NIL Question	#Correct	Precision	#Inexact	#Unsupported
NE identified Question	254	41	16.14%	5	8
Other NE Question	116	5	4.31%	3	2

Table 4.2

5. Conclusion

Of 383 factoid questions whose answer is not NIL, only 46 are correctly answered by our system, so the precision is 12.01%. The accuracy of document retrieval is 71.80% and Bi-sentence retrieval 48.0%. The accuracy of question analyzing, NE recognizing and NE selecting adds up to 34.85%. That is, 65.54% error results from retrieval process including document retrieval and Bi-sentence retrieval, while only 22.45% error results from question analyzing, NE recognizing and NE selecting.

According to our study above, most information lost during retrieval process. So we should focus our further study on seeking better retrieval algorithms, especially Bi-sentence retrieval algorithms.

Reference

[1] M.M.Soubbotin, Patterns of Potential Answer Expressions as Clues to the Right Answer, *In the Tenth Text Retrieval Conference (TREC 10),* 2001.

[2] Rohini Srihari and Wei Li, Information Extraction Supported Question Answer, *In the Eighth Text Retrieval Conference (TREC 8),* 1999.

[3] Amit Singhal, Steve Abney, Michiel Bacchiani, Michael Collins, Donald Hindle, Fernando Pereira, AT&T at TREC-8, *In the Eighth Text Retrieval Conference (TREC 8),* 1999.

[4] Dan Moldovan, Sanda Harabagiu, Roxana Girju, Paul Morarescu, Finley Lacatusu, Adrian Novischi, Adriana Badulescu and Orest Bolohan, LCC Tools for Question Answering, *In the Eleventh Text Retrieval Conference (TREC 11),* 2002.

[5] Hongbo Xu, Hao Zhang, Shuo Bai, ICT Experiments in TREC-11 QA Main Task, *In the Eleventh Text Retrieval Conference (TREC 11),* 2002.