FDUQA on TREC2007 QA Track

Xipeng Qiu, Bo Li, Chao Shen, Lide Wu, Xuanjing Huang, Yaqian Zhou
Fudan University, Shanghai, China, 200433

1. Introduction

In this year's QA track, we only participant in the main task[Dang 2006]. There are two changes in this year. One change is that time dependent questions are added, and the other is that the corpus is consisted by two collections with different qualities. Therefore, we need add some time limitation in answer filter and merge the answers from two different datasets.

The preprocess step is same as our system in TREC QA 2006[Zhou et al. 2006]. We firstly index the documents for fast retrieval. The search engine used in our system is Lucene, an open source document retrieval system. We build four different indexing files. The first two are indexed based on the whole document and the single paragraph of original articles respectively. The rest two are indexed based on the whole document and single paragraph of the morphed articles. Before analyzing question, we process the questions with our question series anaphora resolution.

Our modifications mainly are done for factoid questions and definition questions. For list questions, we used the system in TREC 2006[Zhou et al. 2006]. The only modification is that we used a natural paragraph as a unit to index instead of three sentences.

For factoid questions, we added query expansion and time filter to our system.

For definition questions, we integrate the language model and syntactic features to rank the candidate sentences, and remove the redundancies on sub-sentence level.

The rest of the paper is arranged as follows. Section 2, 3 describe our system of factoid and definition questions respectively. Section 4 presents our results in TREC 2007. At last, we give our conclusions in section 5.

2. Factoid Questions

The framework of our factoid component remains the same as previous years. We resort to the web as the main knowledge resource to find the answer of the question, and then project them to the new Aquaint2 corpus.

Our factoid component includes four main modifications this year: time constraint, query expansion module, a new answer ranking module and answer projecting.

2.1 Time Constraint

Since Trec2007 introduced the concept of time dependency, there may be multiple answers to be extracted for one question without time constraint. If the tense of the question is present, it indicates that the event, which includes the correct answer, should occur recently. So we need find the newest answer for the question.

We analyzed the tense of the question and roughly divided question into two categories.

The first one includes the questions with present tense. Although its time constraint is not stated explicitly, the question in this category seeks for the newest answer. For example, the question is about the present chairman of some organization. For this problem, we use Google to find correct answers because Google tends to prefer the new materials or documents. For the questions like “Who is the chairman of WWF?”, the name of recent chairman of WWF will appear more times than his predecessor in the return list of Google.

The second one includes the questions with time constraint. The time is stated like “in 1993”. For the questions of this category, we assume that the publishing date of the support documents should not be earlier than the time stated in question sentence. Besides, it is not allowed that the time appear in support sentence is different with that in question.

For the rest questions which do not fall into the above two categories, we don’t perform any additional operations.

2.2 Search Module

In the searching phase, a sequence of queries is generated from strict to loose. This strategy was also used in previous years and performed well. In this year, query expansion is added to this phase.

We use the automatic feedback relevance method in the procedure of query expansion. First, we retrieve some relevant documents from the

Web via Google. Second, we extract the terms which are highly relevant to original question. The relevance is calculated by $r(t) = \sum_{t_i \in T} \frac{C(t \& t_i)}{C(t \| t_i)}$, where term *t* is the keyword in original question, *T* is the collection of relevant terms of *t*, which is consisted of the terms around *t* in returned snippets by Google. We constrain that the distance between *t* and the relevant terms is no more than 3. $C(t \& t_i)$ is the count of co-occurrence of term *t* and t_i. $C(t \| t_i)$ is the count of occurrence of either *t* or t_i.

The expanded queries are added to the query sequences and are used as the first query to search the web. The expanded queries can not only improve the recall of the answer, but also increase the average occurrences of the correct answers.

2.3 Ranking Module

In answer ranking, we use a new method to evaluate the answers candidates which are extracted from web. The score of each answer candidate is calculated as follow:

$$Score = s1 * occur + s2 * s\_doc + s3 * s\_sentence + s4 * s\_path, \quad (1)$$

where

occur is the count of occurrences of the candidate in the returned snippets of web search.

s_doc is the score of documents, which is calculated by overlap of keyword and target.

s_sentence is the score of the sentence.

s_path is the score to measure the distance between answer and keyword in the parsing tree. The dependency parse tree is generated by miniparser.

$$s\_path = avg \sqrt{\frac{\sum w_i}{\sum (w_i * dist_i)^2}}, \quad (2)$$

where $\sum w_i$ is the sum of weight of the keyword occurred in the sentence, and $dist_i$ is the distance from the candidate to keyword in the parsing tree. The distance means how many nodes we need go through to reach the

candidate from one specified keyword.

Because the answer candidate may be extracted in several sentences, the final *s_path* is got by calculating the average. w_i is calculated from original question and $\sum w_i = 1$. Generally, the noun is given the largest weight, then verb, number and adjective. And the head word of the phrase is given larger weight than the modifiers.

We tried some machine learning method to tune the parameter in equation (1), such as logistic regression and SVM classifier. But neither methods yield better result than empiric parameter. In our system , parameters are finally set to 0.5, 0.1, 0.1, 0.3. The occurrence score made the biggest impact and the *s_path* score is second.

2.4 Projecting Module

Trec2007 includes two big corpuses to answer question: Aquaint2 and Blog. The projection module in our system is changed a little to adapt to the integration of these two corpuses.

Blog corpus is cleaned by simple strategy. First, we try to remove spam and advertisement links with some features. For example, these advertisement often appear as list of hyperlinks, and their html codes are such as <tr><a>spam here</tr>. Second, all html tags are removed. The remaining text is indexed using Lucene.

We first try to project the answer to aquaint2 corpus. Basically, we believe aquaint2 is a better resource than Blog. If one or more good support documents are found, the projection is done. Otherwise, we turn to Blog for support documents. If no support documents are found in this step, nil will be output for this question.

3. Definition Questions

For definition questions, we first obtain the candidate sentences, and then we integrate the language model and syntactic features to rank the candidate sentences, and remove the redundancies on sub-sentence level.

3.1 Candidate Sentences Generation

We use the question target as query and submit it to retrieval engine. Then we get at most 200 related documents. For each document, we check

all sentences in the document with two simple rules. If no noun word of the sentence appears in the target, or the sentences have more than 70% overlap words with one of the sentences we have extracted, we abandon the sentence. In training phase, the sentences retrieved are used as train samples. In test phase, the sentences retrieved are spitted into short snippets according to the splitting regular expression *"(,|-|) "* and all snippets length should be more than 40. Then, we take all combination of continuous snippets as candidate answer sentences. After applying the learned ranking model, candidate answer sentences are ranked. Then we check redundancies of the candidate answer sentences in turn, and take those as the final answer if they pass the check of the redundancies conditions.

3.2 Feature Extraction

We use features of 3 categories, the first category is based on language models[Zhai 2004, Cui 2004, Cui 2005, Han 2006, Chen 2006], the second is based on syntax of the sentence, and the last contains only one feature, the score of the document returned from IR engine.

3.2.1 Features based on Language Models

To a candidate sentence $s = w_{1,n}$, we take as the different features, $\log P(s \mid \text{Corpus})$, for different corpus. Here we use four corpuses: AQUAINT、processed AQUAINT（AQUAINT*）、definition corpus（DC）和 Target corpus（TC）.

AQUAINT

We train the language model on the collection AQUAINT+AQUAINT2, and calculate the probability *P(s| AQUAINT)* of sentence s, to measure the complexity of *s*。

AQUAINT*

We find that the named entities and numbers in sentence are often related to target, so we replace the person name, location name, organization name with (PRN, LCN, ORG) in the collection AQUAINT+AQUAINT2. We also replace the number with label CD. We calculate *P(s| AQUAINT*)* after the same replacement process with *s*.

Definition Corpus (DC)

We collect the corpus related to target from wikipedia to train the language model. We also process the named entities and numbers in sentence like AQUAINT*. Sine this corpus is small, we do Dirichlet smoothing on AQUAINT*. *P(s| DC)* is the probability that *s* is a definitional sentence.

Target Corpus (TC)

We use target as queries and submit it to Google，we collect the first 100 returned snippet as target corpus. Similarly，we do Dirichlet smoothing on AQUAINT. *P(s| TC)* is measuring the relatedness between *s* and target.

Thus, we get the four features of the sentence *s, log P(s| AQUAINT), log P(s| AQUAINT*), log P(s|DC),* and *log P(s|TC)* based on language model.

3.2.2 Features based on Syntax of a Sentence

We use Minipar to analyze each sentence, and get a set of triples {w1，rel, w2}. For any relation *rel-a*, if there is a triple (w1, rel-a, w2), where one of w1, w2 is not stop word and appears in target, another is not in target, we define *rel-a(s)*=1, else *rel-a(s)*=0, and the relation *rel-a* is used as a feature. However, All relations do not help to find the correct answer. We use chi-square test to select four features, which are the punctuation *"punc"*, the appositive *"appo"*, the complement clause of prepositional phrase *"pcomp-n"* and the grammatical subject *"s"*.

3.3 Removing Redundancy and Getting Final Answer

Algorithm 1　Algorithm of Removing Redundancy

Initialize a word pool *WP* as empty set
$i \Leftarrow 1$
while length(*FA*) < threshold and *i* < number of candidate sentences **do**
$x \Leftarrow i$th in the candidate sentences
if *R(x,WP)=0*
Add all words of *x* into *WP*
Take *x* as part of the final answer *FA*
endif
endwhile

To the ranked candidate answers, we check the redundancy from the

top. Algorithm 1 shows the detailed process, where *FA* is final answer set and *WP* is a word pool maintained in the process, *R(x,WP)* is used to indicate whether *x* is a redundant and is calculated as *R(x,WP)=1* if 70% of the words of one of the snippet of *x* are in the *WP*, and 0 else.

3.4 Differences among the 3 submitted results

In the run1, sentences of trec2007 targets are retrieved from Aquaint2. The difference of run1 and run2 is that, in the run2, sentence selection is based on the whole sentence and the step of removing redundancy is not used. In the run3, sentences are retrieved from Aquaint2 and BlogCorpus and the Topic Corpus (TC) in feature extraction is defined according to the target types.

4. Evaluation

We submitted three runs for the main task of TREC15 QA Track: FDUQAT16A, FDUQAT16B and FDUQAT16C.

Table 1 Evaluation Results of FDUQA Runs in TREC QA 2007

		FDUQAT16A	FDUQAT16B	FDUQAT16C	Best	Mean	Worst
Factoid Question	**Accuracy**	0.236	0.228	0.228	0.706	0.131	0.019
List Question	**Average F score**	0.107	0.131	0.101	0.479	0.085	0.000
Other Question	**Average F score**	0.291	0.329	0.309	0.329	0.118	0.000
Final Score		0.213	0.231	0.215	0.484	0.108	0.015

From this table, we can see that we get some improvements of our factoid and other question answering systems. Moreover, the algorithm we use to answer definition questions is quite promising.

5. Conclusions

In this year, we focus our attentions on factoid and other question, and get some improvements which mainly are derived by adding the syntactical features. However, there're still a lot of things to be improved in our question answering systems. Some more sophistic methods can be used to improve the performances.

6. Acknowledgements

This research was supported by the National Natural Science Foundation of China under Grant No. 60435020. We are very thankful to Zhongchao Fei, Feng Ji, Xiaofeng Yuan, Yindong Chen and Wei Pan for their contributions in our works.

7. Reference

Hang Cui, Min-Yen Kan, Tat-Seng Chua, Jing Xiao. A comparative Study o Sentence Retrieval for Definitional Question Answering. In Proceedings of the 27th Annual International ACM SIGIR Conference, 2004

Cui, H., Kan, M.-Y., and Chua, T.-S. Generic soft pattern models for definitional question answering. In SIGIR05: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, New York, NY, USA, 384-391, 2005

Chen, Y., Zhou, M., and Wang, S. 2006. Reranking answers for definitional QA using language modeling. In Proceedings of the 21st international Conference on Computational Linguistics and the 44th Annual Meeting of the ACL, Morristown, NJ, 1081-1088, 2006

Han, K., Song, Y., and Rim, H. 2006. Probabilistic model for definitional question answering. In Proceedings of the 29th Annual international ACM SIGIR Conference on Research and Development in information Retrieval (Seattle, Washington, USA, August 06 -11, 2006). SIGIR '06. ACM Press, New York, NY, 212-219.

Hoa Trang Dang. Overview of the TREC 2006 Question Answering Track. In Proceedings of the 15th Text Retrieval Conference (TREC-2006), pages 54-68, 2003

Zhai, C. and Lafferty, J. A study of smoothing methods for language models applied to information retrieval. ACM Trans. Inf. Syst. 22, 2 (Apr. 2004), 179-214

Yaqian Zhou, Xiaofeng Yuan, Junkuo Cao, Xuanjing Huang, Lide Wu. FDUQA on TREC2006 QA Track, Proceeding of the 15th Text Retrieval Conference, Gaithersburg, USA, 2006