FDUQA on TREC2004 QA Track

Lide Wu, Xuanjing Huang, Lan You, Zhushuo Zhang, Xin Li, Yaqian Zhou
Fudan University, Shanghai, China, 200433
Email:{ldwu,xjhuang,lan_you,zs_zhang,lixin,zhouyaqian}@fudan.edu.cn

1 Introduction

In this year's QA Track, we process factoid questions in a way that is slightly different from our previous system [1]. The most significant difference is that we developed a new answer type category, and trained a classifier for answer type classification. To answer list questions, we use a pattern-based method to find more answers other than those found in the processing of factoid question. And an algorithm that uses some knowledge bases answers definition questions. This algorithm achieves a promising result.

In our system, external knowledge is widely used, which includes WordNet and Internet. The ontology in WordNet is used in the answer type classification, and its synsets are used to do query extension. Internet is used not only to find factoid question answers, but also as knowledge base for definition questions.

In the following, Section 2, 3, 4 will separately introduce our algorithm to solve factoid, list and definition questions. Section 5 will present our results in TREC2004.

2 Factoid Question

2.1 Flow Chart

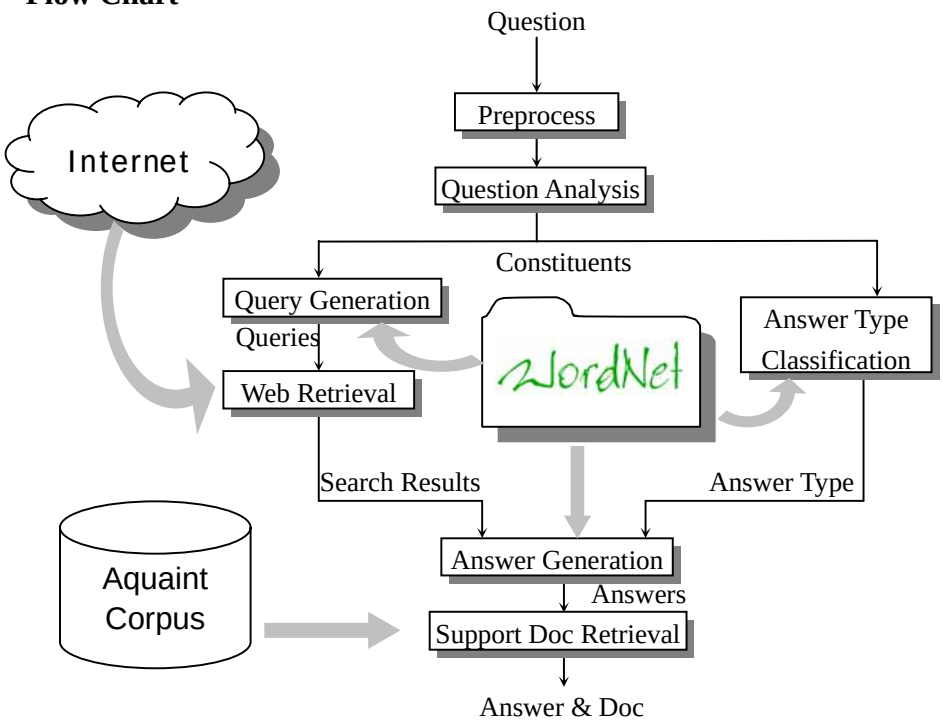

Figure. 1. Flow Chart of FDUQA on Factoid Question

The above figure describes the process of our factoid question answering. Details about each module will be introduced in the latter subsections.

2.2 Question Preprocess

In TREC2004, a set of questions is about a certain target so that anaphora is used in questions. For example, target 1 is "Crips", question 1.5 is about it: “What is their gang color?” In this question, “their” refers to “Crips’”. So each question should be preprocessed first to replace these pronouns with their references. A simple algorithm is used here to solve this anaphora. We replace the pronoun and some short form like “the group”, “the comet” etc. with the question’s target. Then in the following, we can process questions in a traditional way.

2.3 Question Analysis

We do question analysis by LinkParser [2], an English parser based on link grammar. In this step, constituents are extracted from the question sentence. For example: “When was the Hale Bopp comet discovered?” Its constituents are: “the Hale Bopp comet” – subject; “was discovered” – predicate. This constituents information, as we will see in the following, is used in answer type classification, query generation, as well as answer generation.

2.4 Answer Type Classification

In the answer type classification step, FDUQA system determines the answer type of the input question based on some ordered list of rules obtained by machine learning. We adopt a thirty-one-class answer type classification system, illustrated in table1. Different types of the questions are treated differently in the following answer generation module.

Besides POS and interrogative, WordNet is used as an external resource in our question classifier. Meantime, constituents of question sentences obtained in parsing above are used too. They are crucial for classification processes.

Table 1. Answer Type concepts

NAMEBASIC	PRN	LCN	ORG
PIECEOFWORK	QUOTATION	POSTADDR	ABBR
TIM	DAT	NUMBASIC	NUMBER
ORDINAL	AGE	MEASUREBASIC	LENGTH
PCT	MNY	INTEGER	CODEBASIC
URL	TELEPHONE	POSTCODE	EMAILADDRESS
BNP	DESP_OF_ABBR	TRANSLATION	MANNER
REASON	CONSEQUENCE	OTHER	

The classifier and focus words decision algorithm are both based on Transformation-Based error-Driven learning (TBL) [3]. TBL is an approach to corpus -based natural language processing. By using TBL, a rule-based question classifier is developed to determine the question focus and answer type.

Firstly, we use TBL to get the focus in the question. The question focus is a word or a phrase, which indicates what the question is looking for, or what the question is all about. It emphasizes on the type of answer expected. Especially, it should be existed in WordNet, or else there is no focus in the question.

An example of a rewrite rule for focus: *Change the tag from 'NN_0 to 'NN_1'*
And an example of its triggering environment is: what $$ WP_0 NN_0.

This rule means if the interrogative of a question is 'what' and following word is a noun, the noun will be tagged as a focus candidate. When all the transformations have been used, we get the focus of the input question.

Secondly, we search the Wordnet and get the focus word's sense. All its Hypernyms in WordNet and other information are input into another TBL classifier, and then we get the category of a question.

An example of a rewrite rule for category: *Change the tag from '_#' to _TIM'*
And an example of its triggering environment is: *what nil $$ clock time, time $$ _#*

This rule means if the interrogative of a question is 'what', the verb of question is not 'be' (nil) and the question do not have a category (_#) also, but the focus is a Hyponyms of *clock time, time.* The question has the candidate category TIM. After every transformation is used, we will get the final category of a input question.

The questions of TREC8-11 are used as training set, and TREC12 questions are used as test. We get about 700 rules for focus and 500 rules for category. They are used to the input questions one after another. The precision of answer type classification over questions of TREC12 is 84.2%.

2.5 Query Generation and Web Retrieval

We find answers from Internet using Google search engine. Query generation is subject to the characteristics of Google Search such as phrase search. To utilize this characteristic, we combine constituents in such way:

In the last example, "When was the Hale Bopp comet discovered?" has constituents: "the Hale Bopp comet" – subject, "was discovered" – predicate. Queries for this question are: "the Hale Bopp comet was discovered" and "the Hale Bopp comet" "was discovered".

Upon the basic queries, we also do some query extension. Here are some rules for this extension:

- Synonym extension – to replace the noun in the query with its synonym, which is found in

WordNet.

- Preposition extension – to add some prepositions to queries when the question asks for location or time.
- Unit extension – to add some units to queries when the question asks for measure.

2.6 Answer Generation and Support Document Retrieval

Answer generation is based on the answer type category. We first extract candidate answers in snippets returned by Google using an NE tagger. Then each answer will be given a score according to its context. Now we obtain a list of answers from Internet. The support documents are retrieved in Aquaint Corpus with the answers obtained and the question target as queries. By measuring the distance between the answer and key words, we will get the best answer and its support document.

3 List Question

This year, we use patterns to extract answers for list questions. The answers of a list question always appear in a paratactic structure like "… and …". We obtained these structures from the past TREC list questions, and built a knowledge base for them.

Here are some examples from our knowledge base:

P1. (including|include|included|includes|involve|involving|involves|involved) (<A>)+ and <A>

P2. such as (<A>)+ and <A>

P3. between <A> and <A>

P4. (<A>)+ and <A>

P5. (<A>)+ as well as <A>

These patterns are expressed in regular expression. <A> here means the answer. The answer found by factoid question processing module is used as a seed in pattern matching.

For example: "Name cities that have an Amtrak terminal."

We first treat it as a factoid question and answer "New York" was found. Then we use "New York" as a seed that must appear at <A> in patterns.

Now we find the following context match the pattern "(<A>)+ and <A>" with the seed of "New York":

We found the following sentence matched the pattern: Preliminary plans by Amtrak that were released yesterday call for stops of its high-speed express service in *Boston , Providence , R.I. , New Haven , Conn. , New York , Philadelphia , Baltimore and Washington* .

So "Boston", "Providence", "R.L.", "New Haven", "Conn.", "Philadelphia", "Baltimore", "Washington" will be extracted. And then we validate whether these phrases are noun phrases as "New York". If yes, they will be put into answer list.

This algorithm depends not only on patterns, but the seed answer as well. If the seed answer is wrong, it's almost impossible to find right answers.

4 Definition Question

As for answering definition question, we develop a method based on online knowledge bases. Many question answering systems have showed the importance of resource on the Web for answering factoid questions [4], and the results of our system reveal that the Web-based knowledge bases are also quite helpful to answering definition questions.

The flow chart for definition question of FDUQA is in figure2.

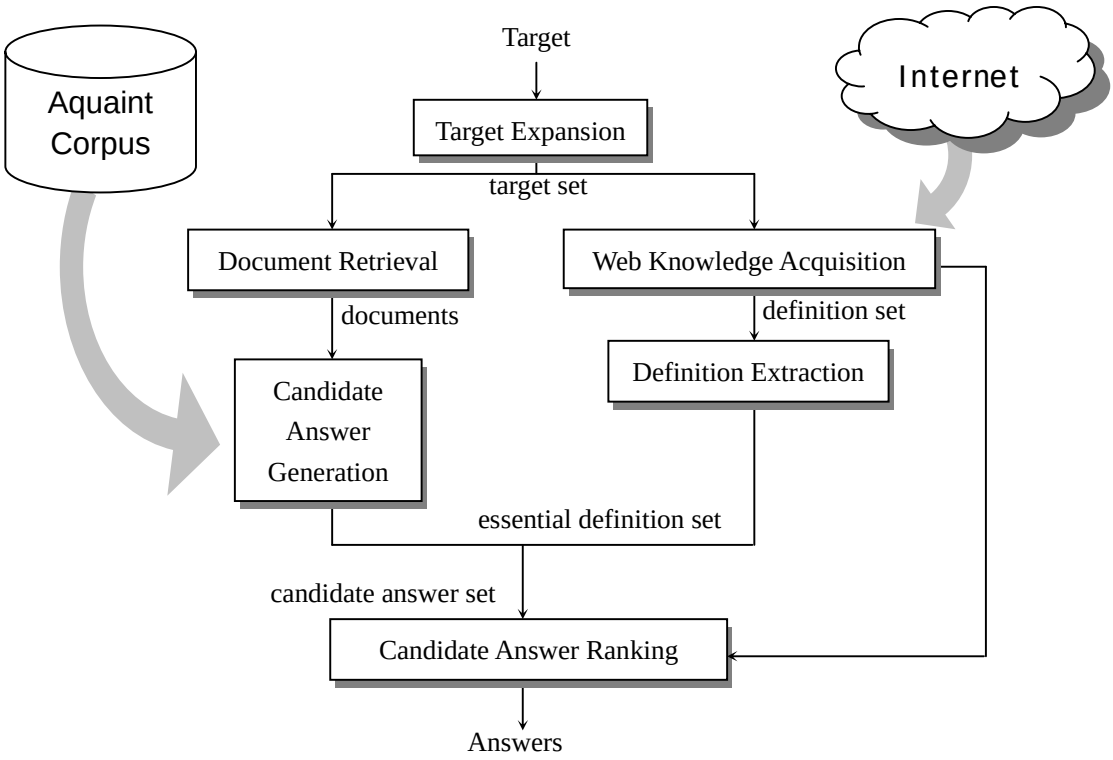

Figure2. Flow Chart for Definition Question of FDUQA

4.1 Target Expansion

Usually, a target to a definition question may have several different forms. We use synsets in WordNet to find them and expand the “target” into a “target set”. For example, the target “Khmer Rouge” can be expanded into {“Khmer Rouge”, “KR”, “Party of Democratic Kampuchea”, “Communist Party of Kampuchea”}. This expansion can make us both retrieve more in the corpus and get more knowledge from knowledge bases.

4.2 Document Retrieval and Candidate Answer Generation

We get the documents about the target in the AQUAINT corpus. Search engine OKAPI is used for this purpose. For a question, each element in the “target set” was submitted to the search engine as a query one time. All the search results are kept for candidate answer generation later. There are about 100 documents for each question.

Then we cut the documents into sentences, and delete the sentences that have no relation with the target using a few heuristic rules. For example, we delete all the sentences that contain no element of the "target set". We also delete the redundancy. For two arbitrary sentences, if their content words overlap above 65%, we drop one of them. The sentences remained form the candidate answer set, which can be expressed as $S_A\{A_1, A_2, \ldots, A_m\}$, where A_k (k=1..m) is a candidate answer in the set and m is the total number of the candidate answer..

4.3 Web Knowledge Acquisition and Definition Extraction

We first search the definitions about the target from a number of online knowledge bases. These knowledge bases are the WordNet glosses and other online dictionaries such as the biography dictionary at www.encyclopedia.com. For each target, these definitions from different knowledge bases form its "definition set", which is expressed as $S_D\{D_1, D_2, ..., D_n\}$, where D_k (k=1..n) are definitions about the target from different resources, and n is the definitions' number.

Different definitions in the "definition set" may overlap in some degree, and one definition may have some information that is not very important. So we extract the essential information from the "definition set" to form the "essential definition set". We hope that this set may cover all the most important information about the target and little redundancy. The extraction is based on a summary technique. $S_{ED}\{d_1, d_2, \ldots, d_l\}$ is the "essential definition set" to the target, where each element d_i is an essential information about the target.

4.4 Candidate Answering Ranking

In this module, we have tried two methods to rank the candidate answer set. They use the "definition set" and the "essential definition set" separately.

4.4.1 Ranking with Definition Set

Let S_{ij} be the similarity of D_i and A_j, where the similarity is the tf-idf score, with D_i and A_j treating as a bag of word. We get the score of each candidate answer A_i as follows:

$$score_i = \sum_{j=1}^{n} w_j S_{ij} \quad (\sum_{j=1}^{n} w_j = 1)$$

The weights w_j are fixed based on the experiment on TREC2003 definition question set. Rank the set S_A based on $score_i$, and choose the top ones as the answer. This method is used in the run FDUQA13a and FDUQA13b with slightly different weights.

4.4.2 Ranking with Essential Definition Set

FDUQA13c was generated using this ranking method.

The following algorithm was executed to get the answer set of a definition question:

1. Initially set the answer set A={}

2. For each element d_i (i=1..*l*) in the set S_{ED}: we first get the similarity between d_i and A_j(j=1..m), which is expressed as S_{ij} and calculated as the same as in 4.4.1. Then find the candidate answer in the set S_A as follows: if $S_{ik} = \max\{S_{i1}, S_{i2}, ..., S_{im}\}$ and $S_{ik} > minsim$, then add A_k to the set A and delete A_k from the set S_A.

3. If $\sum_{k=1}^{m'} L(A_k) > \max\_answer\_length$ or no element is found in step 2, the algorithm ends; otherwise, go to step2, where $L(A_k)$ represents the length of string A_k in character and m' is the number of elements in set A.

The parameters *max_answer_length* and *minsim* were empirically set based on TREC12's 50 definition questions. The last result set is A $\{A_1, A_2, \ldots, A_{m'}\}$.

The performance of this module can be found from Table 2 in section 5. The F-measure score of the former ones are slightly higher than the latter. The simple method may get the better result.

However, the result of the latter one is also encouraging. Since it has the potential to get the answers, which are not only brevity but also have wide coverage about the target. Of course it depends on the quality and the coverage of the essential definition set. We believe that the performance may be boosted after improving the extracting method of essential definition set, and we will try it in the future.

We think that the methods presented give an appropriate framework for answering definition questions.

5 Results

We submitted three runs for the main task of TREC13 QA Track: fduqa13a, fduqa13b and fduqa13c. In these three runs, the algorithms used to answer factoid questions are merely different in some parameters. The results of list questions in the runs are just the same. As to definition questions, difference between these three runs is described above.

Table 2. Performance of FDUQA Runs in TREC13

		Fduqa13a	Fduqa13b	Fduqa13c
Final Score		0.265	0.262	0.256
Factoid Question	#correct	59	59	59
	#unsupported	30	27	27
	#inexact	4	5	5
	#wrong	137	139	139
	Accuracy	0.257	0.257	0.257
	Precision of NIL	2/12 = 0.167	2/17 = 0.118	2/17 = 0.118
	Recall of NIL	2/22 = 0.091	2/22 = 0.091	2/22 = 0.091
List Question	Average precision	0.143	0.143	0.143
Definition Question	Average F score	0.404	0.389	0.367

From this table, we can see that the algorithm we use to answer definition questions is quite promising. The list questions are answered not so well. It's maybe due to the patterns we use are not enough, and the correctness of seed answers is doubtable.

Acknowledgements

This research was partly supported by NSF of China under contracts of 60103014, as well as the 863 National High-tech Promotion Project of China under contracts of 2001AA114120. We are very thankful to Yongping Du, Ningyu Chen, Feng Ji, Feng Lin and Jian Huang for their contributions in our work.

Reference

[1] Lide Wu, Xuanjing Huang, Yaqian Zhou, Yongping Du, Lan You.2003. *FDUQA on TREC2003 QA Task*. Proceedings of the TREC-12

[2] Daniel Sleator and Davy Temperley. 1993. *Parsing English with a Link Grammar.* Third International Workshop on Parsing Technologies.

[3] Brill, Eric *Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging.* Computational Linguistics, Dec. 1995

[4] Jimmy Lin. *The Web as a resource for question answering: Perspectives and challenges*. In Proceedings of the Third International Conference on Language