UNT 2005 TREC QA Participation: Using Lemur as IR Search Engine

Jiangping Chen, Ping Yu, He Ge

School of Library and Information Sciences
University of North Texas
P.O. Box 311068, Denton, TX 76203
{jpchen, hg0022, pingyu}@unt.edu

Abstract: This paper reports our TREC 2005 QA participation. Our QA system EagleQA developed last year was expanded and modified for this year's QA experiments. Particularly, we used Lemur 4.1 (http://www.lemurproject.org/) as the Information Retrieval (IR) Engine this year to find documents that may contain answers for the test questions from the document collection. Our result shows Lemur did a reasonable job on finding relevant documents. But certainly there is room for further improvement.

1. Introduction

Question Answering (QA) aims at identifying answers to users' natural language questions. A QA system can release the users from digesting large amount of text in order to locate particular facts or numbers. Therefore the research is much needed and has drawn great attention from several disciplines such as information retrieval, information extraction, and artificial intelligence.

TREC QA track has provided comparable QA system evaluation on sets of test questions since 1999. The degree of difficulty of the test questions has increased substantially in recent years, which push the research toward applying more sophisticated strategies and better understanding of English texts.

Question answering is very challenging due to the ambiguity of the questions, complexity of linguistic phenomena involved in the documents, and the difficulty to understand natural languages. A QA system typically contains multiple functional modules in order to find the answers from a large text collection. It takes a team several years of hard work in order to build an effective QA system. Our prototype QA system, named EagleQA, made use of available NLP (Natural Language Processing) tools and knowledge resources for question understanding and answer finding. We skipped information retrieval (IR) process in 2004 because we were overwhelmed in building the basic QA modules.

This year's QA track required that each team also submit a list of documents that were used by the QA systems to find the answers in addition to the answers themselves. We started to consider expanding our current QA system to include a module for document retrieval.

Lemur becomes one of our candidates for IR search engines among others, such as Smart (ftp://ftp.cs.cornell.edu/pub/smart/) and Lucene (http://lucene.apache.org/java/docs/). The Lemur Toolkit (http://www.lemurproject.org/) is designed and developed by researchers from the Computer Science Department at the University of Massachusetts and the School of Computer Science at Carnegie Mellon

University. The project is sponsored by the Advanced Research and Development Activity in Information Technology (ARDA) and by the National Science Foundation (NSF). There are several reasons to consider Lemur, including:

a. Lemur supports document indexing and multiple well-known text retrieval models such as the TFIDF retrieval model, the Okapi BM25 retrieval function, and the InQuery (CORI) retrieval model (http://www.lemurproject.org/lemur/retrieval.html). Other systems focus on only one method.
b. The developer states that the toolkit is "under constant development for performance improvements as well as feature additions" (http://www.lemurproject.org/news.html), which we feel is a desired feature. Also, the forum provides quite good technical support.
c. The system is designed as a research system, and is quite convenient to be used for TREC-type IR experiments because it accepts TREC document format and produces TREC-type results for evaluation.
d. The toolkit is expandable and adaptable with available source codes. We can adapt Lemur for various purposes such as Cross-Language Information Retrieval and Question Answering.

We applied Lemur 4.1 to find relevant documents for 2004 QA test questions and found it returned more relevant documents for most of the questions than NIST search engine. Lemur was also used in our other experiments [2] [3]. Therefore, we decided to use Lemur as the search engine for this year's QA experiments.

This paper describes the overall structure of our QA system, NLP tools and lexical resources employed by EagleQA, our QA methodology for TREC 2005, QA and document retrieval test results & analysis, and our plan for future research.

2. System Overview

Current EagleQA system is comprised of 7 major modules or subsystems:

a. *Question Processing*. Accept users' questions and performs part-of-speech tagging, phrase bracketing, keyword identification & expansion, and answer type identification.

b. *Document Retrieval*. Apply Lemur 4.1 to find relevant documents for each question from the AQUAINT document collection.

c. *Document Annotation*. Apply LingPipe (http://www.alias-i.com/lingpipe/) and Minipar (Lin, 1994) to annotate English texts. LingPipe is used first to detect sentence boundaries, the identified sentences are sent to Minipar for part-of-speech tagging and named entity categorization. LingPipe can also perform named entity categorization and co-reference annotation. At last, we integrate the results of annotations from the two systems using an XML format.

d. *Sentence Retrieval*. Identify 500 non-duplicate sentences from the annotated documents as sentence candidates which may contain an answer to each test question. The keywords and answer types obtained in Question Processing are utilized to find matched sentences for each factoid and list question. For "Other"

questions, the sentence retrieval subsystem returns the sentences that match the target as answer candidates.

e. *Web QA*. Google.com (http://google.com) is utilized to find possible candidates of answers.

f. *Answer Finding*. Look for multiple evidences to identify and determine answers for factoid and list test questions. Factors that are taken into account when ranking an answer candidate include: 1) answer type; 2) weight of the sentence; 3) distance to keywords in the same sentence; and 4) whether it is a candidate returned by Web QA.

g. *Answer File Formulation.* Combine the answers for different types of questions such as factoid, list, and other questions into a submission file. It also removes duplicate answers from the list of answers to other questions if the answers are selected for any factoid or list questions for the same target.

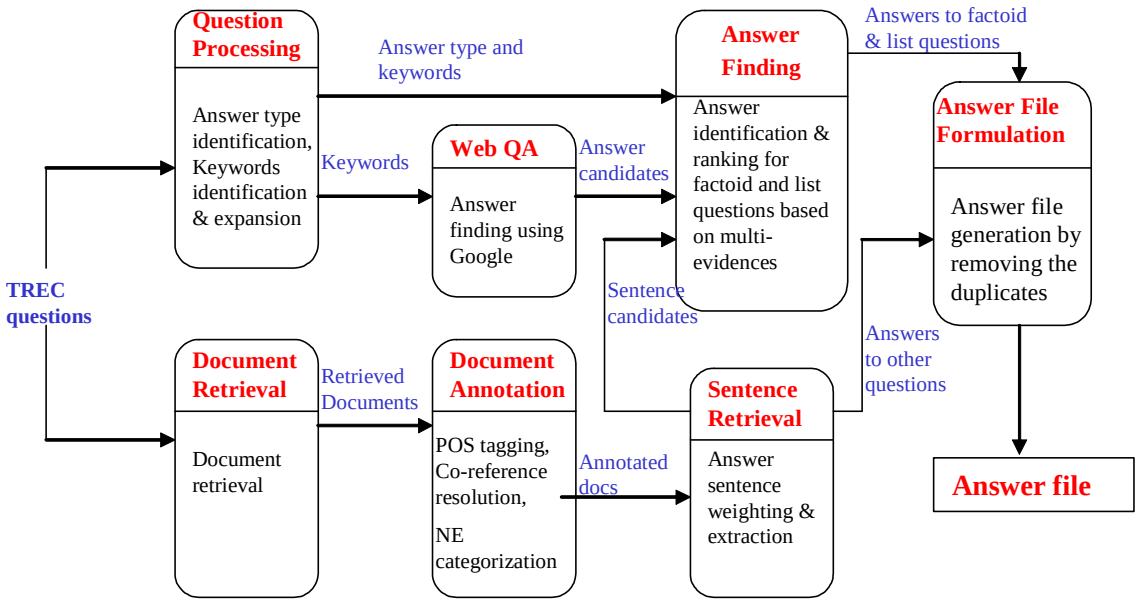

Figure 2. EagleQA Architecture

Figure 1 is the system architecture of current EagleQA. Compared to last year's system, Document Retrieval module has been added and Web QA module has been modified to test a new method. Below we will further describe these two modules. Descriptions for other modules were included in our 2004 TREC paper [1].

2.1 Document Retrieval

Document retrieval using Lemur is straight forward as Lemur has been intentionally designed for TREC-type information retrieval experiments (http://www.lemurproject.org/lemur/indexingfaq.html). The AQUAINT document collection was first indexed by Lemur using a simple manually created stop word list

including only articles, pronouns, and propositions. To form the query file for document retrieval, we added the target to each Factoid and List question, and replaced 'other' in the Other questions using their targets. Then we converted the questions into the format required by Lemur for retrieval. Our experience in other document retrieval experiments using Lemur [2][3] demonstrated that Okapi BM25 retrieval model produced the best performance. Therefore, we only applied this model to return the retrieved documents for all our three QA runs. The default setting for Okapi BM25 was applied. The official results of document retrieval are reported in Section 5.

2.2 Web QA

The Internet is a huge and unique knowledge base. Our Web QA subsystem attempts to make use of this knowledge resource by submitting the original test questions to *Google*. The top 100 short summaries returned by Google are sent to the Documentation Module for annotation. Then the annotated texts were sent to Answer Finding Module to locate a list of answer candidates (about 20). Those answer candidates can later be used by Answer Finding module as a weighting factor. The above strategy was different from what we had done for TREC 2004. For TREC 2004 evaluation, we wrote a simple program concerning only the frequency of each term (a word or a phrase) and its category as the factors of answer candidate identification and ranking. This year, we want to apply a consistent approach to answer finding no matter the retrieved texts come from the Web or the predefined document collection.

3. NLP tools and Knowledge Resources

EagleQA makes use of many freely available (for research purposes) natural language processing (NLP) software systems and knowledge resources in various modules or subsystems in order to find answers for the test questions. Table 1 lists all the tools and knowledge resources used by our system.

Table 1. Incorporated NLP tools and knowledge resources

Applications	URLs if Obtained Online	Modules that Use the Application	Usage Description
Lemur IR Toolkit	http://www.lemurproject.org/	Document Retrieval	English document indexing and retrieval
LingPipe	http://www.alias-i.com/lingpipe/	Document Annotation	English sentence boundary detection, named entity annotation
Minipar	http://www.cs.ualberta.ca/~lindek/minipar.htm	Document Annotation	English Part-of-Speech tagging, information extraction, noun phrase annotation
WordNet	*(http://www.cogsci.princeton.edu/~wn/*	Question Processing, Answer Finding	Synonym and Hyponym extraction
WordNet::QueryData	http://people.csail.mit.edu/u/j/jrennie/public_html/WordNet/	Question Processing, Answer Finding	Synonym and Hyponym extraction
Google.com	http://www.google.com	Web QA	Finding answer candidates from the Web

4. QA Methodology for 2005

Our QA strategy this year has not changed much from last year. We were not able to implement new strategies even though we have realized that EagleQA needs to be improved. In general, we employed similar strategy to Factoid questions and List questions except that a threshold was applied to List questions in order to determine the number of answers that should be returned. The threshold was predetermined based on experiments using previous years' test questions. Other questions are quite different from Factoid questions. Therefore we used a different approach to answer them.

4.1 Factoid questions and List questions

Factoid questions and List questions were handled following the procedures of question processing, document retrieval, document annotation, sentence retrieval, Web QA, answer finding, and answer file formulation. EagleQA first performed a shallow co-reference resolution to replace pronouns in questions with the actual targets. For example, in last year's question "*Which was the first movie that he was in?*" "*he*" was replaced by the target "*James Dean*". Then we used Minipar to tag each question and WordNet to expand nouns and verbs in the question. Final result for each question includes the answer type and a list of keywords with expansions if any. For example,

Question: *Which was the first movie that James Dean was in?*
Answer type: movie
Keywords: *movie(synonym: movie, film, picture, moving picture, moving-picture show, motion picture, motion-picture show, picture show, pic, flick) / James Dean*

After annotating the retrieved documents using LingPipe and Minipar in the Document Annotation module, the annotated texts were sent to Sentence Retrieval module to identify sentences that may contain answers to the test questions. Then the sentences were processed by Answer Finding module to identify the answer for each question. The description of sentence retrieval and answer finding was included in our last year's TREC paper.

4.2 Other questions

The other questions were different from Factoid and List questions. Other questions require QA systems to provide information about the targets in addition to answers to the factoid and list questions for the same targets. We applied the same strategy as last year to find answers to Other questions. The only change is the threshold we used to cut off number of answers that should be returned for each Other question.

5. Results & Analysis

We submitted three runs this year, namely UNTQA0501, UNTQA0502, and UNTQA0503 for the main task. Their official scores are listed in Table 1. Following describes the differences on strategies of the three runs.

- UNTQA0501. This run did not use the results from the new Web QA module. The answers were solely based on answer finding from the AQUAINT collection;

- UNTQA0502. This run did not use the results from the new Web QA module either. But a different answer type pattern file that includes patterns extracted from last

year's test questions was applied by the Question Processing module to process List questions;

- UNTQA0503. This run used the results from the new Web QA module. It also applied the new answer type pattern file as UNTQA0502 to process List questions.

All the above three runs used the same document retrieval results returned by Lemur. Table 2 presents the document retrieval evaluation results for the 50 questions specified by NIST.

Compared to last year, EagleQA did worse this year for all the three types of questions. Its performance on factoid question is even a little bit lower than the median. The scores for the other two tasks are only a little bit higher than the medians. The unsatisfactory performance may due to the increase of the degree of difficulty in the test questions. But obviously, current system needs careful evaluation and big effort for improvement.

The three runs do not show significant difference even though the strategies were different on certain modules.

Table 1. Official QA Results

Run	Factoid (Accuracy)	List (Average F)	Other (Average F)	Average per-series score
UNTQA0501	0.135	0.054	0.191	0.131
UNTQA0502	0.135	0.064	0.182	0.132
UNTQA0503	0.141	0.062	0.184	0.134
median	*0.152*	*0.053*	*0.156*	*0.123*
Best	*0.713*	*0.468*	*0.248*	*0.534*

Table 2. Official Document Retrieval Results

Run	Relevant Document	Retrieved Relevant Document	Average Precision	R-Precision
The 50 questions in ALL UNT Runs	1575	841	0.3285	0.3205

Document retrieval using Lemur can return about 53.5% relevant documents. We feel this is a reasonable performance. Lemur can find at least one relevant document for 88% of the test questions (44 out of 50).

6. Future Research

Still, our QA system EagleQA is at a very early development stage. We could not carry out further testing and development due to our work on other tasks such as cross-language question answering for NTCIR-5 this year. Fortunately, we have more time to work on EagleQA in 2006 and we plan to test something new on answer finding.

Our current answer finding strategy is problematic. The factors that are contributed to answer candidate identification are limited due to a lack of a semantic parser. The

ranking strategy for answer candidates needs training and tuning. The Answer Finding module will be the focus of our QA research in 2006.

The retrieval performance of Lemur will be further analyzed and we hope to find ways to improve document retrieval using this system.

7. References

Chen. Jiangping; Ge, He; Wu, Yan and Jiang, Shikun. (2004). UNT at TREC 2004: question answering combining multiple evidences. Online Proceedings of TREC 2004. Available at: http://trec.nist.gov/pubs/trec13/papers/unorthtexas.qa.pdf.

Chen, Jiangping; Li, Rowena; Yu, Ping; Ge, He; Chin, Pok; Li, Fei and Xuan, Cong. (2005). Chinese QA and CLQA: NTCIR-5 QA experiments at UNT. Proceedings of NTCIR-5 workshop, Tokyo, Japan, December 2005.Available at: http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings5/data/CLQA/NTCIR5-CLQA-ChenJ.pdf.

Chen, Jiangping; Li, Rowena and Li, Fei. (2005). Chinese information retrieval using Lemur: NTCIR-5 CIR experiments at UNT. Proceedings of NTCIR-5 workshop, Tokyo, Japan, December 2005. Available at: http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings5/data/CLIR/NTCIR5-CLIR-ChenJ.pdf