11-791 Homework 2 Report

Yuanchi Ning
Andrew ID: yuanchin **Date:** Sep. 2013

1. Design

1.1 Goal

Given the goal that designing and implementing the logical architecture and UIMA analysis engines of specific input artifacts (which are text files here) for question-answering purpose, the given type system (containing annotation types of `Question`, `Answer`, `Token`, `NGram` and `AnswerScore`) is used to further design and build the system.

1.2 System Design

Since the type system given is almost the same as the self-designed type system that I did in the last assignment, here the type system (logical data model) is completely adopted from which the `deiis_types.xml` describes.

Based on the type system, different annotation methods are designed and are briefly described as follows:

Algorithms of Generating the Annotations		
Annotation Type	**Corresponding Algorithm**	**Brief Description**
Question	TestElementAnnotator	Parse the text spanned in the document making use of the beginning "Q" character of each line.
Answer	TestElementAnnotator	Parse the text spanned in the document making use of the beginning "A" character of each line. And record the correctness of each `Answer` by parsing the second column in each answer line, where 1 indicates correct and 0 indicates incorrect.
Token	TokenAnnotator	Parse the text spanned in the `Question` and `Answer` annotations. Use the punctuation and space to split the sentence, and further get tokens.
NGram	TokenUniGramAnnotator TokenBiGramAnnotator TokenTriGramAnnotator	Parse the text spanned in the `Question` and `Answer` annotations, making using of `Token` annotations already generated by checking the consecutiveness of the tokens. Generate the 1-Gram, 2-Gram and 3-Gram separately.

AnswerScore	AnswerScoreAnnotator	Using the N-Gram overlap scoring method to score each `Answer`, which is the ratio of total number of 1-Gram, 2-Gram and 3-Gram in `Question` overlapping with the `Answer` to the total number of 1-Gram, 2-Gram and 3-Gram in `Answer`.

When parsing the Token annotator, it is better to add a stemming step on each token so that when comparing the overlap between the Question and the Answer later, the matching proportion is more accurate. However, this needs to using extra natural language analysis dependencies, which I haven't figured out how to link it with UIMA framework yet. Therefore in this design, the basic algorithm described above is still adopted.

When deciding the scoring method, token overlap and NGram overlap scoring methods are considered. For a question answering system, wrong answers are commonly found in the form of using a negative form of auxiliary verbs or link verbs. Therefore the token overlap scoring method usually assigns a very high score to this kind of answers falsely. And in this case, the NGram overlap scoring method relatively performs better. Better scoring method should take the semantic meaning of the question and answer sentence into consideration, and not only based on the statistic characteristics. But implementation of this angle goes beyond my capability for this assignment, so here just uses the NGram overlap method for scoring.

2. Implementation

2.1 Primitive Analysis Engine

As described above, each annotator processing algorithm is corresponding to a java class file that implements the annotating method. And the several primitive analysis engines are built for each of the java file as the basic components for further analysis.

Primitive Analysis Engine Descriptor	
Descriptor	**Output**
TestElementAnnotator	Question or Answer Annotator
TokenAnnotator	Token Annotator
TokenUniGramAnnotator **TokenBiGramAnnotator** **TokenTriGramAnnotator**	NGram Annotator
AnswerScoreAnnotator	AnswerScore Annotator
Evaluator	The summary of the predicted results (containing the precision of each question and the average precision of the whole collection)

2.2 Aggregate Analysis Engine

Given the primitive analysis engine listed above, the final aggregate analysis engine can be organized using these elements in a fixed flow to analyze the questions and answers.

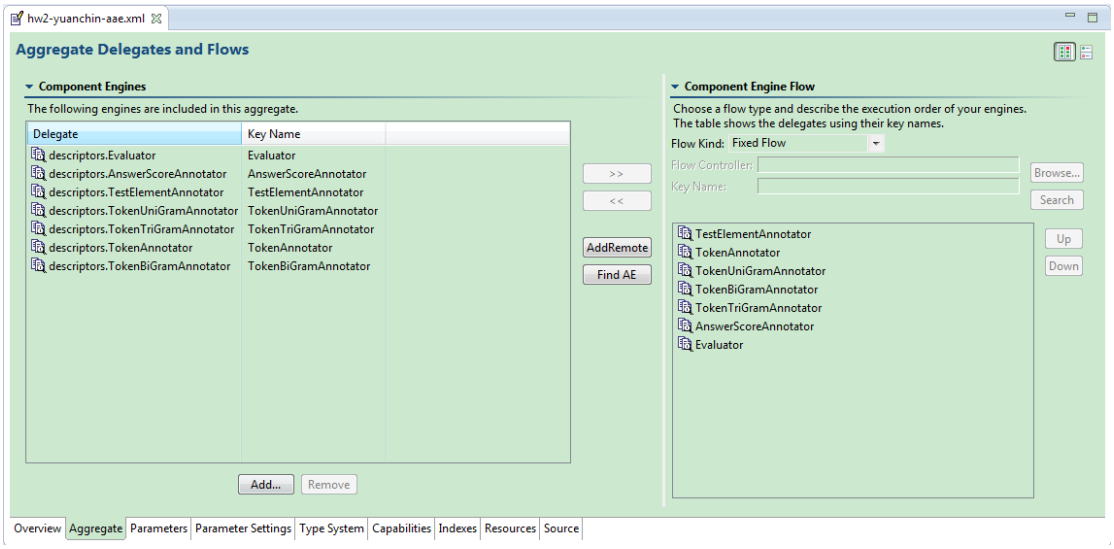

Fig 1. Aggregate Analysis Engine Descriptor in Implementation

2.3 Experiment Result

The final evaluation of the predictions outputted from the console is depicted as below:

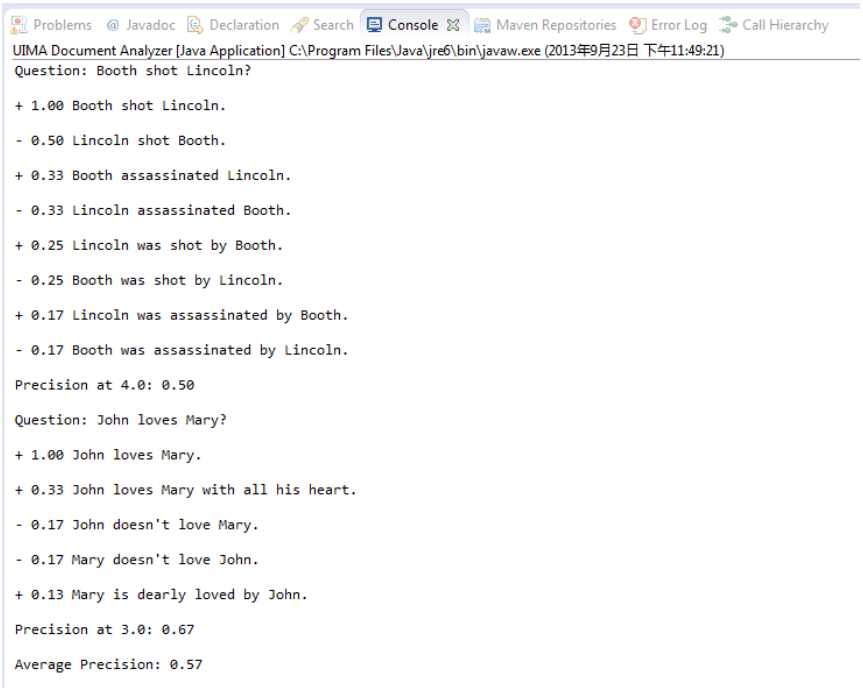

Fig 2. The output results.

Summary

This assignment is a real implementation of the former assignment and there is lots of flexibility in designing the system as well as choosing the scoring method. However when considering of analyzing in this task, if we want to implement more complicated scoring methods, I found that it's difficult to make a balancing between the accuracy of the results and the generality of the method. This should be a very interesting and meaningful aspect for question answering system for further research.