11-791 Homework 3 Report

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

1. Design

1.1 Goal

Given the goal that creating the CPE, building an UIMA-AS client to call a remote service, and deploying own UIMA-AS service and then testing it, this assignment is focused on the learning and implementation of these processes. In addition, this report discusses some extra incorporation with annotations from Stanford CoreNLP.

1.2 System Design

For Task 1.2, the system design is simple. Based on *the Evaluator annotator* from the last homework assignment, a new CAS consumer has been created. And the integrated CPE descriptor performs just as same as the last assignment performs.

For Task 2.2, a new remote analysis engine has been created to integrate the remote UIMA-AS service Stanford CoreNLP. By adding this remote analysis engine into the aggregate analysis engine pipeline in the homework 2, and then integrating it into the CPE pipeline, the *Stanford CoreNLP Annotator* has been utilized in this assignment. Based on the new annotations that the remote service provides, more information can be used to calculate the answer score of each answer more accurately. Here the *Name Entity* annotation has been add into the answer scoring component, and the *Token POS Tagging* feature has been experimented to score the answer as well.

By compare the *MentionType* and the text of *Name Entity* in the *Question* and *Answer*, a NameEntity-Overlapping scoring method similar to NGram-Overlapping has been integrated into the original scoring method. When utilizing the Token POS Tagging in answer scoring, the score based on two features of *Token* annotation, *POS* and *Lemma*, has been integrated into the answer score.

For Task 2.3, the deployment part does not include many design stuff. In the testing part, an analysis engine describing my own service has been created and has been integrated into the CPE pipeline to use the service.

2. Implementation

2.1 Basic Implementation Notes

The implementation flow of this assignment is conceptually simple. However, the actual implementation process is tough, since lots of weird problems have been caused during the process and none of them are covered in the tutorial.

By trying adding additional dependencies, re-building the project and re-updating the Maven projects and a lot of other testing methods, these problems encountered have finally been solved.

2.2 Answer Scoring Part

In the answer scoring part of this assignment, different scoring methods have been experimented.

In addition to the NGram-Overlap scoring method used in the last assignment, some new scoring method based on the new annotations generated by the Stanford CoreNLP Annotator have been integrated into the original score.

$$\text{NewScore} = \frac{1}{3}(\text{NGramOverlapScore} + \text{NameEntityOverlapScore} + \text{TokenPOS\&LemmaOverlapScore})$$

When calculating the NameEntityOverlapScore, the percentage of Name Entity annotations in Answer having the same MentionType and same Text with the Name Entity annotations in Question has been used.

When calculating the TokenPOS&LemmaOverlapScore, the percentage of Token Entity annotations (generated by the remote service) in Answer having the same POS Tag and the same Lemma with the Token annotations in Question has been used.

2.3 Experiment Result

Comparison of the runtime cost of each task is shown as below:

Task	Basic CPE	Remote Service	Own Service
Time (s)	1.249	1.496	2.936

From the time costs shown above we can see that the time cost of using the remote service is higher than using the local analysis engine, which satisfies our expectation. And since the scale of input data isn't very large, the difference in runtime is not very large.

The final evaluation of the predictions outputted from the console can be compared as below:

(Precision)	NGram	+Name Entity	+Name Entity&Token
Input 1	0.5	0.5	0.5
Input 2	0.67	0.67	0.33
Average Precision	0.57	0.57	0.43

Although after adding the Token POS and Lemma part into the answer score the precision gets lower, it should be caused by the characteristics of the input data set. Taking the order of the tokens into consideration, or analyzing the *Relation Annotation* generated by the Stanford CoreNLP Annotator should achieve better scoring prediction. However in this assignment these considerations haven't been tested yet.

Summary

This assignment really makes me clear about how to utilize a remote service, how to deploy my own service and how to use the UIMA-AS to achieve a better application of asynchronous and scaleout scenarios. During the implementation, learning the tutorial and exploring methods to fix unexpected weird problems cost a lot of time. But analyzing the new annotations generated by the remote service and testing different scoring method is interesting.