Assignment 3
11-791: DEIIS

Submitted by: Ankur Gandhe, ankurgan

October 7, 2013

1 Task 1

The task of this assignment is to create an execution architecture with a CPE (collection processing Engine). To create the CPE for out sample information processing system, we created 3 descriptor files

1. Collection Reader Descriptor: Since we were reading the files from our local system, we could use the FileSystemCollectionReader class provided in UIMA. We had to include 'uima-examples' typesystem to be able to read the files. The parameters for input directory, language and encoding were set in the descriptor file.

2. Cas Processors: Cas processors take the CAS files read and process them. In our case, we added the aggregate analysis engine to this. Modifications were made to the aggregate engine from hw2 such that the CAs processor only processed the data and created annotations for answer score, evaluation and precision. It did not output anything to stdout, or write to a file as required by best practices for cas processors.

3. Cas consumer: The cas comsumer takes the processed cas and then uses it to evaluate/display. In our case, we wrote code to print the question, ranked list of answer and the precision score generated by cas processor.

2 Task 2.2

Since task 2.1 was learning UIMA-AS, we skip to task 2.2 where we were required to create a client for the Stanford CoreNLP Annotation from clearTK and use the named entity annotations in our scoring system.
We created a descriptor file that would call the cleartk service deployed at the specified brokerURL and take the named entity mentions as inputs to the AnswerScoreAnnotations. We incorporated the following additional features using cleartk's annotations :

- Named entity mentions: In our answer scoring class, we added code to read the named entities present in the question and match them with the named entities present in the answers. The fine neScore was the number of matches of named entities divided by the total number of named entities in the question

- Lemma : Since Cleartk offers a token annotation which stores lemma of the word (loves $\rightarrow$ love), we thought it might be useful to incorporate this as a feature. Similar to named entity score, we create a lemmaScore to match the lemma's in the question with the answer.

- Negation Penalty: Since the presence of negation in an answer is an indication that the answer might be opposite of what we want, we added a penalty for presence of 'not' in the answer. Again, the not was obtained by looping over the lemma's present in the answer since it converts doesn't to do not.

Using the scores above, and the ngramScore from previous homework, we calculated the final score as the weighted average of the above features.
Below is a table of comparison between current system and homework 2: The time delay in

System	Precision Doc1	Precision Doc2	Avg Precision	Time
HW2	0.5	1	0.75	0.169 sec
Current	0.5	1	0.75	1.072 sec

current system comes from calling StanfordNLP service: it takes 949 sec to be complete.
To better compare the 2 system, we print ranked lists output by the 2 systems below:

```
Q John loves Mary?
HW2:
+ 1.0 John loves Mary with all his heart.
+ 1.0 John loves Mary.
+ 0.33 Mary is dearly loved by John.
- 0.33 Mary doesn't love John.
- 0.33 John doesn't love Mary.

Current:
+ 0.94 John loves Mary with all his heart.
+ 0.94 John loves Mary.
+ 0.6 Mary is dearly loved by John.
- 0.55 Mary doesn't love John.
- 0.55 John doesn't love Mary.
```

As we can observe, there is no change in the precision, but the second system pushed the score of wrong answers more down while the first system could have places the right answer below the wrong one.

2.1 Task 2.3

To deploy our own service. We started a broker on our system via the command line. We then created a deploy descriptor for our pipeline on a queue "QASystem" with the brokerURL of the already started broker. Care was taken to add the jar dependencies to the UIMA_CLASSPATH and then the service was deployed. THe same service was successfully called by writing a client descriptor that called for service from the "QASystem" and the correct brokerURL.
The speed of the system was much slower now (742ms compared with 169ms) but this was expected.

2.2 Task 2.4

We could not find sufficient time to do all the bonus question, but we did try using the lemma provided by Stanford in the answer scoring