11-791 : Design and Engineering of Intelligent Information Systems
Homework 3

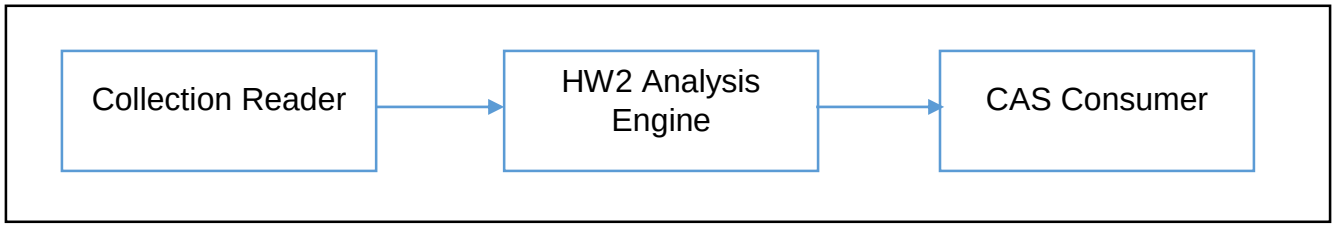
Name : Soumya Batra
Andrew ID : soumyab

Task 1.2:

System Design: We create our CPE with the following 3 main components:

1. **Collection Reader** : reads the input document(s) and stores it/them in CAS
2. **Analysis Engine** : our analysis engine from HW2 with little variations(explianed later). This processes the document cas and creates question, answer and answerscore annotations and stores them in CAS
3. **CAS Consumer** : includes a JAVA class that processes the annotations thrown by the analysis engine and returns a ranked list of answer candidates and calculates individual and overall precision

Flow Diagram: Following is the flow diagram of the order of processing in our CPE :

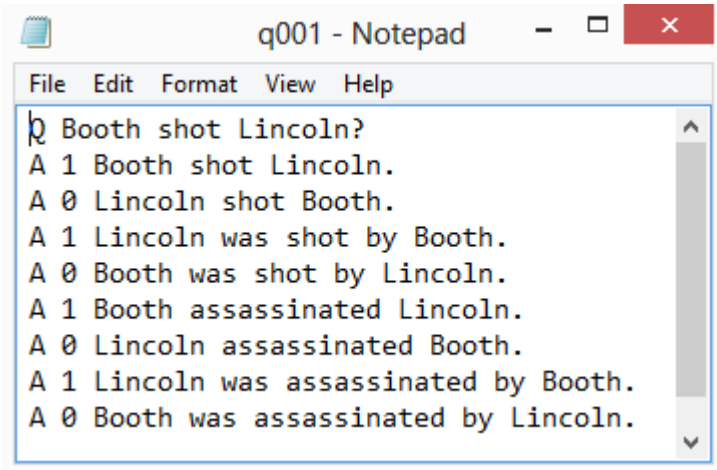


Change in Analysis Engine from HW2: There are two following changes in the currnt analysis engine from HW2 :

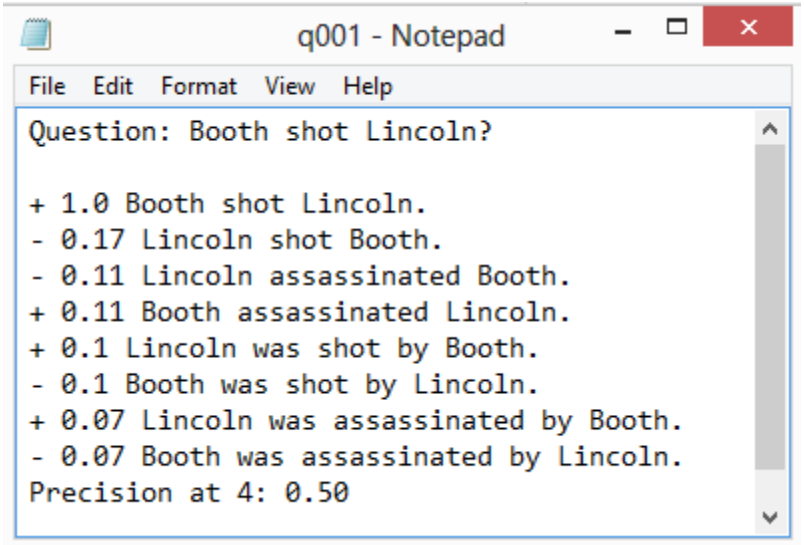
1. We **do not use GoldAnswer scoring** to get a score for the answers and use only NGrams. This is because we wish to compare results after using NamedEntitiy Annotator and if GoldAnswer Scoring method is used, it will always result in the right answers, assuming Gold Answers are correctly identified.
2. **Answer Ranking and Evaluation is** removed from the Analysis Engine. Instead, it is **moved to the CAS Consumer component**. This is done to meet the requirements of Task 1.1.

<u>Results</u> : We give the following two input files to our CPE and get the following results:

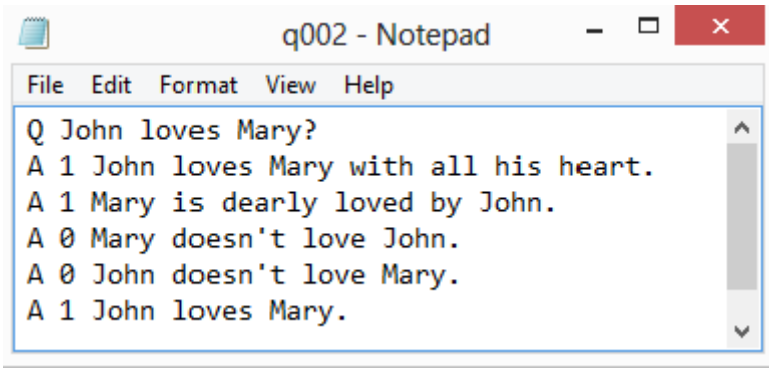
1. **Input** : q001.txt



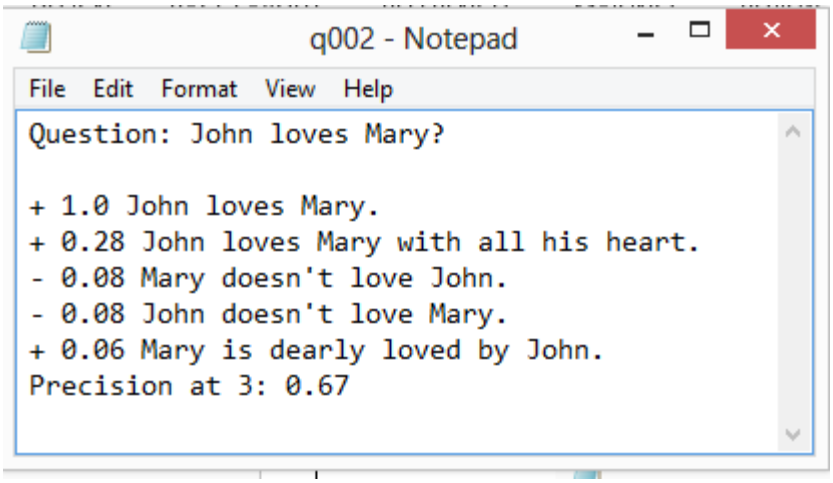
Output : q001.txt



2. **Input :** q002.txt

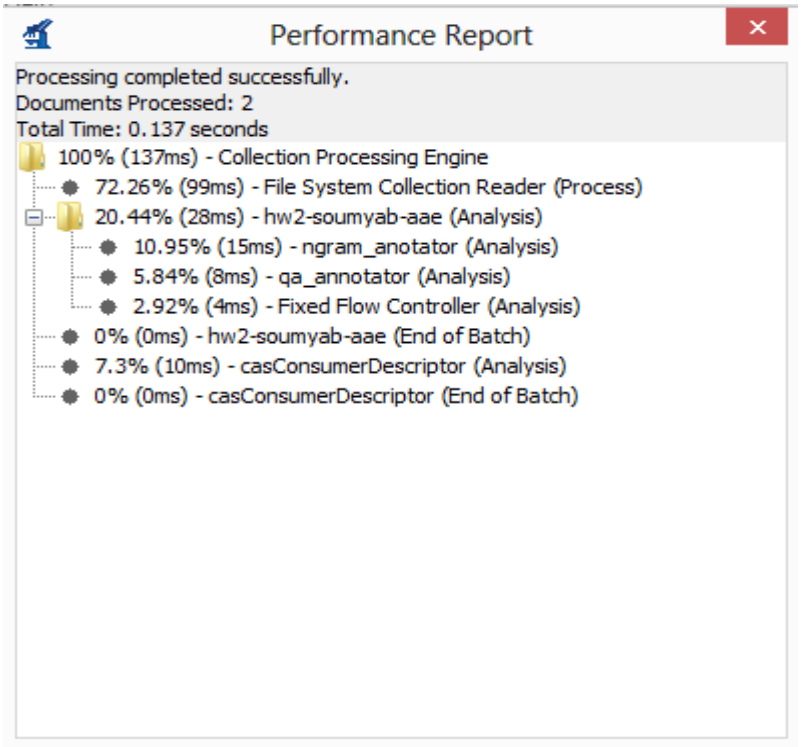


Output : q002.txt



Average Precision (displayed on Console): 0.58

Speed: The entire processing took 0.137 seconds

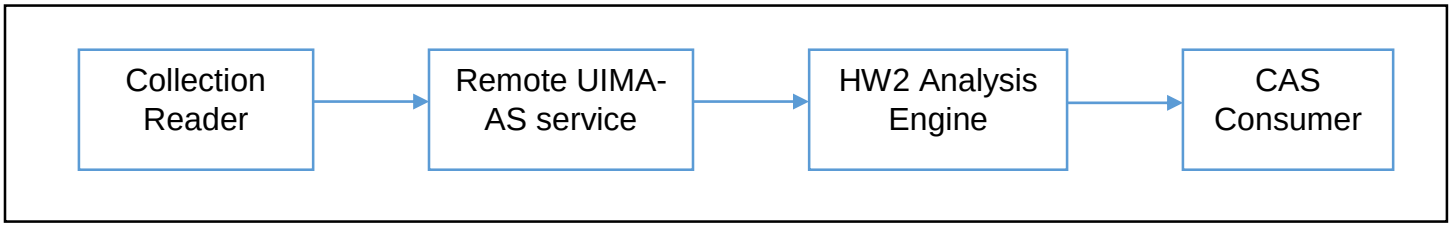


Task 2.2:

System Design: We modify our CPE from Task 1.1 as follows:

1. **Collection Reader** : same as before
2. **Client Descriptor for Stanford Core NLP** : uses remote UIMA-AS service provided by Stanford Core NLP to get NamedEntity annotations. Note that we changed default timeout to 15s instead of 5s.
3. **Analysis Engine** : same analysis engine with added capabilities to update answer scores using named entities
4. **CAS Consumer** : same as before

Flow Diagram: Following is the flow diagram of the order of processing in our CPE :



<u>Results</u> : We give the following two input files to our CPE and get the following results:

1. **Input** : q001.txt

q001 - Notepad

File Edit Format View Help

```
Q Booth shot Lincoln?
A 1 Booth shot Lincoln.
A 0 Lincoln shot Booth.
A 1 Lincoln was shot by Booth.
A 0 Booth was shot by Lincoln.
A 1 Booth assassinated Lincoln.
A 0 Lincoln assassinated Booth.
A 1 Lincoln was assassinated by Booth.
A 0 Booth was assassinated by Lincoln.
```

Output : q001.txt

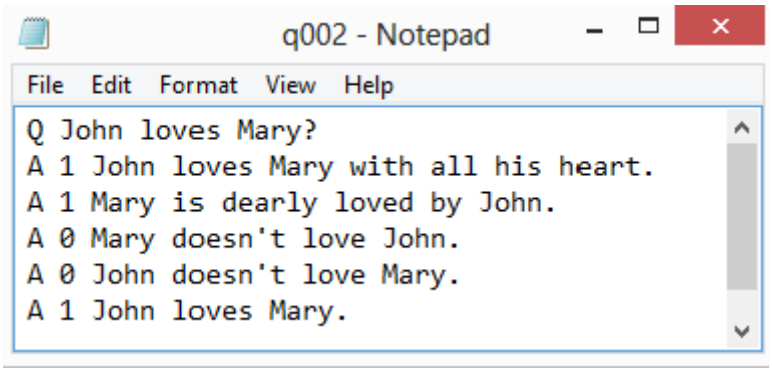
q001 - Notepad

File Edit Format View Help

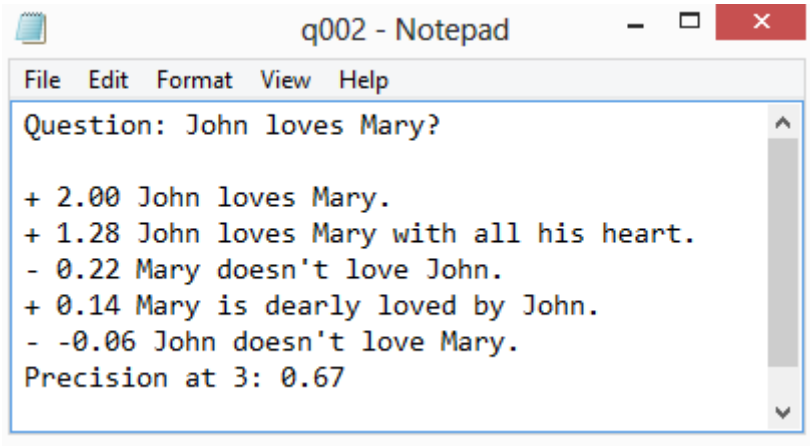
```
Question: Booth shot Lincoln?

+ 2.00 Booth shot Lincoln.
- 1.17 Lincoln shot Booth.
+ 1.11 Booth assassinated Lincoln.
- 1.11 Lincoln assassinated Booth.
+ 1.10 Lincoln was shot by Booth.
- 1.10 Booth was shot by Lincoln.
+ 1.07 Lincoln was assassinated by Booth.
- 1.07 Booth was assassinated by Lincoln.
Precision at 4: 0.50
```

2. **Input :** q002.txt

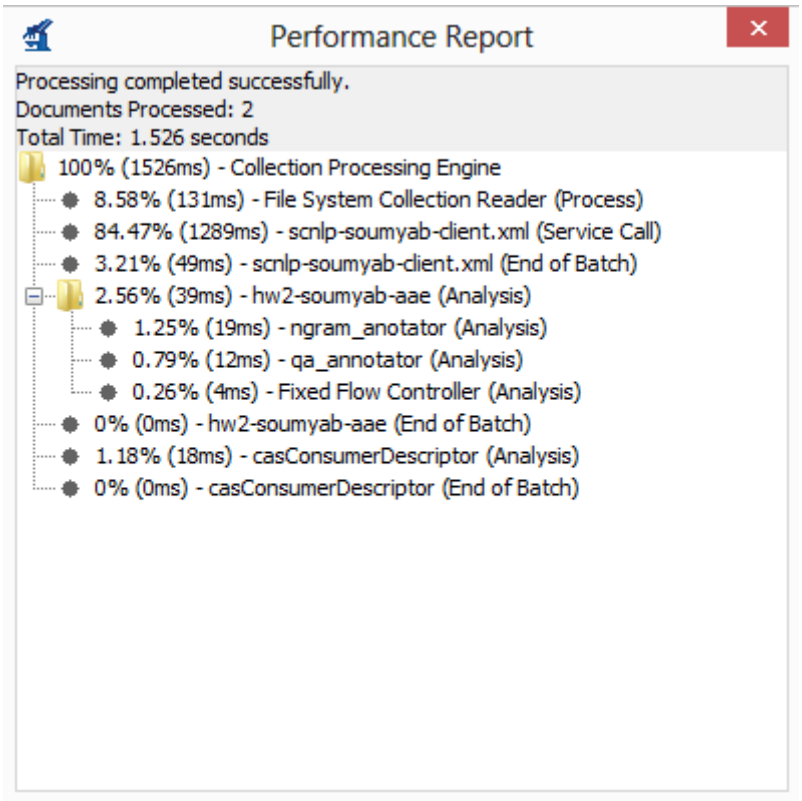


Output : q002.txt



Average Precision (displayed on Console): 0.58

Speed: The entire processing took 1.526 seconds



<u>Comparison with Task 1.1:</u>

1. Average Precision for both tasks is same (=0.58).
2. Task 2.2 took a longer time overall.
3. Even though the overall and individual precision remained the same, using Named Entities brought candidate answers higher by giving them a higher score. For e.g., for 2nd input file, q002.txt, “Mary is dearly loved by John” got bumped up by 1 with a score of 0.14 while the wrong answer “Mary is not loved by John” got weighed down and in fact, got a negative score. It can be easily seen that with higher number of test cases, using Named Entity annotations will produce a better result. Also, improving the answer score by using token overlap, stop wordlist, semantic trees, etc. we will get a better result with Named Entity annotations.

Hence, **Task 2.2 has better precision but lower speed than Task 1.1**.

Task 2.3:

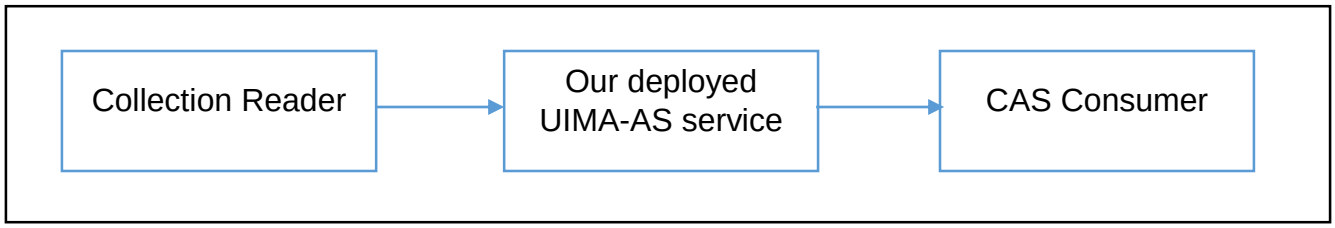
Service Deployment: We deploy our Analysis Engine as follows:

1. Create a deployment descriptor for the Analysis Engine.
2. Start the broker locally from command line by calling startBroker.bat file.
3. Deploy the Analysis Engine from Eclipse by running as 'UIMA Deploy AS Service' and selecting the Deplyment Descriptor XML file and adding project src folder to the Source.

Testing the Service: We test the above deployed service as follows:

1. Create a client descriptor for the service as in task 2.2.
2. Create a CPE descriptor for the service that contains the following 3 components:
   2.1 Collection Reader : same as before
   2.2 Client Descriptor for our UIMA-AS service
   2.3 CAS Consumer : same as before

Flow Diagram: Following is the flow diagram of the order of processing in our CPE :



Results : Same as before. The service runs with a console output:

"Average Precision: 0.58".