

11-791 Design & Engineering of Intelligent Information Systems

Hw2-Report

AndrewID: Chongshm

1. The Design Aspects

1.1 UML Design Diagram

The Design Pattern, which I used as the following UML diagram, I wrote five classes to deal with the gene nametag NER problem.

The File system collection reader will help the system to read the file line by line. Then, the three different analysis engines could process the data from the reader. By integrating the three different annotators, the performance was improved significantly.

The last step, the consumer will receive the cas from the ConfidenceAnnotator. And write the information from the cas into the ouput document.

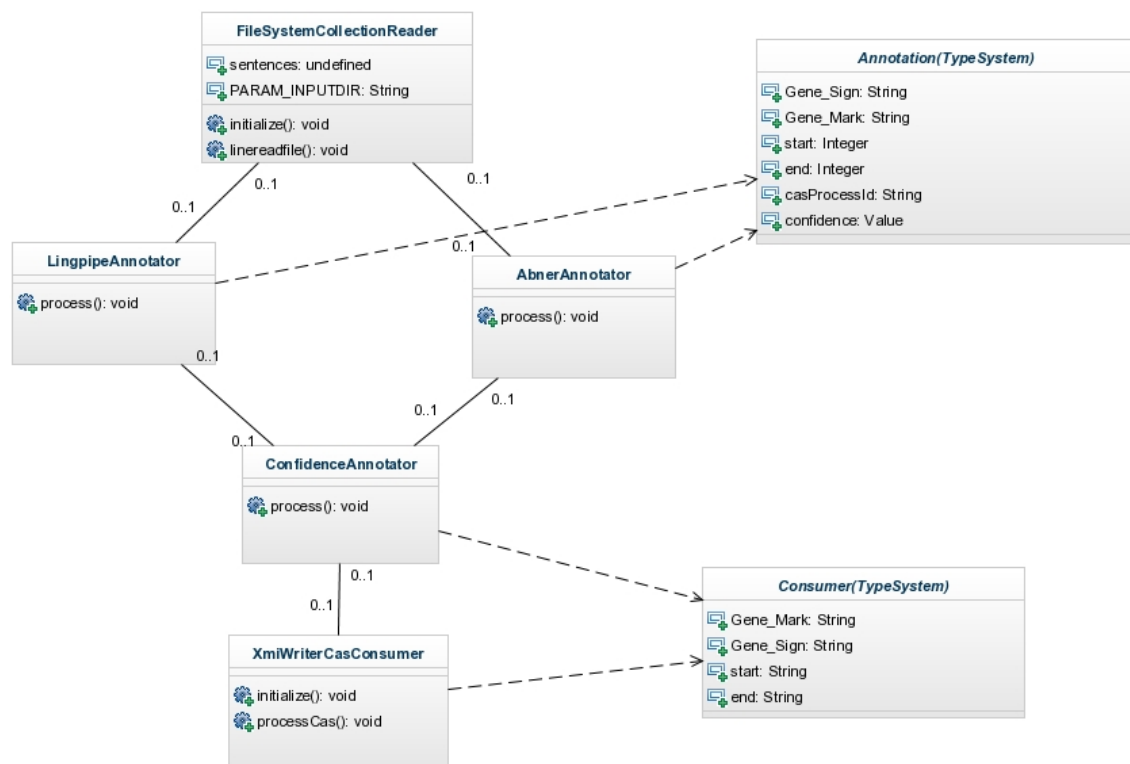


Figure 1. The UML diagram of gene NER

1.2 Design of File Collection Reader:

As same as the homework 1, the pattern of reading file in the FileSystemCollectionReader.class also used the line-by-line, then save each line as a JCas. Thus, these JCas will move to the next step for Analysis Engine.

1.3 Type System Design:

I used two different type systems in this homework. The type system, which is called the Annotation, I create it with 4 features as following:

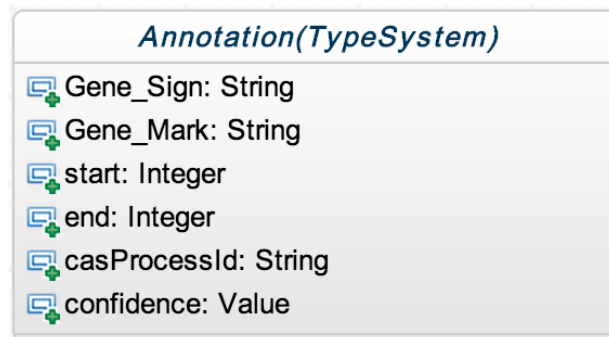


Figure 2. The first type system of the first two annotators.

- (1)Gene_Mark, which store the gene tag retrieved by Annotator.
- (2)Gene_Sign, which represents the ID of this Gene.
- (3)start, which is the start position of the possible Gene name tag.
- (4)end, which is the end position of the possible Gene name tag.
- (5)casProcessorId, which is used for identifying which annotator give the recognizing result.
- (6)confidence, which store the annotator confidence of the potential gene tag.

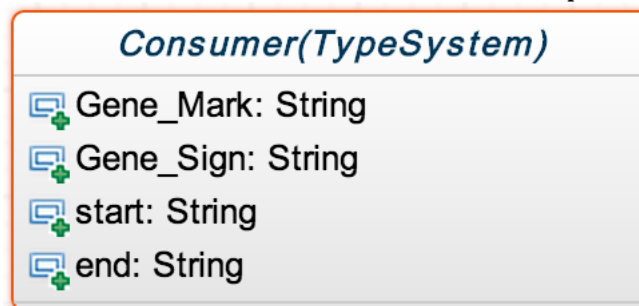


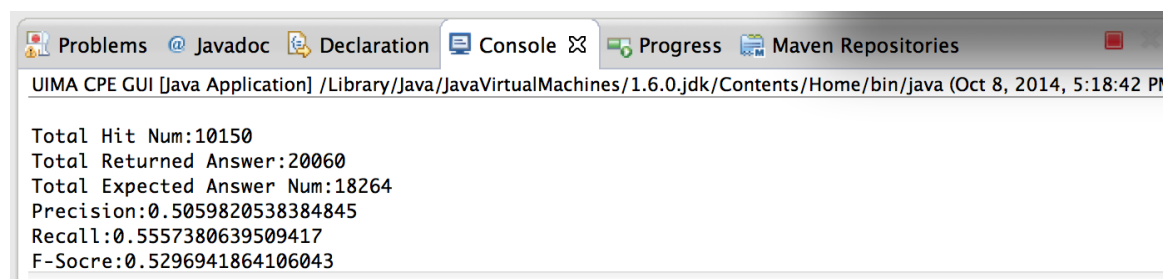
Figure 3. Anthor type system for Confidence Annotator and CasConsumer.

After finishing producing all the potential gene tag, the output results should be filtered. Therefore, I created another the type system to store the results after filtering. Then, store the filtered information in this type system, Consumer. The CasConsumer will process these cases whose information are stored in this type system.

- (1)Gene_Mark, which store the gene tag retrieved by Annotator.
- (2)Gene_Sign, which represents the ID of this Gene.
- (3)start, which is the start position of the possible Gene name tag.
- (4)end, which is the end position of the possible Gene name tag.

1.4 The Design of Analysis Engine Structure

AbnerAnnotator: At ABNER's core is a statistical machine learning system using linear-chain conditional random fields (CRFs) with a variety of orthographic and contextual features. The link of this API is <http://pages.cs.wisc.edu/~bsettles/abner/>. Using the Abner API to process the input data, then extract the potential Gene tags that are considered as the answer. The result of using Abner alone:



```
UIMA CPE GUI [Java Application] /Library/Java/JavaVirtualMachines/1.6.0.jdk/Contents/Home/bin/java (Oct 8, 2014, 5:18:42 PM)

Total Hit Num:10150
Total Returned Answer:20060
Total Expected Answer Num:18264
Precision:0.5059820538384845
Recall:0.5557380639509417
F-Socre:0.5296941864106043
```

Figure 4. The result of using Abner alone.

LingpipeAnnotator: In the homework 1, the lingpipe was used for the main Analysis Engine for dealing with the input, and its performance could meet the demand. And, in the first time, I utilized the First-Best Named Entity Chunking function to recognize the possible gene tag. However, in this task, the First-Best Named Entity Chunking could not perform well as I expected. Then, the Confidence Named Entity Chunking function came into my eye. Meanwhile, this function could also generate a confidence value for each possible gene nametag according to its algorithm.

The result of using Lingpipe alone:

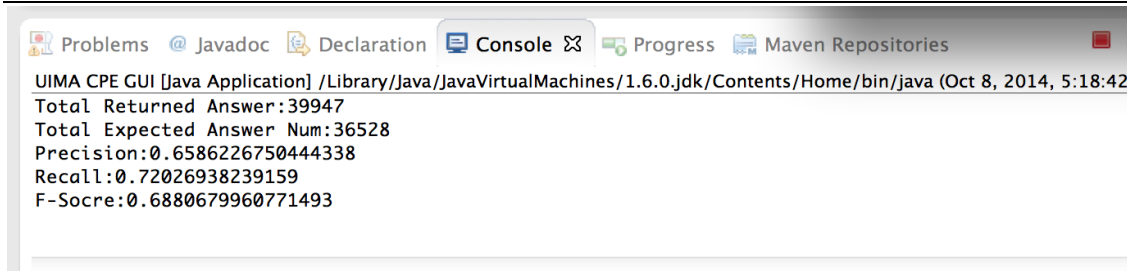


Figure 5. The result of using Lingpipe alone(First-Best).

ConfidenceAnnotator: Owing to the unsatisfied result from using these two annotators separately, there should be a new way to filter the results that are returned by these two annotators. In order to deal with the problem that extracts the most precise data from two analysis engines, we need to compare the result from the Abner API and Lingpipe API. Based on testing these two analysis engines respectively, I noticed that the output from Lingpipe could be seem as the better choice for the final result. Owing to the higher precision of Lingpipe, for each gene ID, I intended to believe the result of Lingpipe NER, rather than the Abner NER. But, when the confidence of Lingpipe is non-ideal, we will try to output the result of Abner NER. Moreover, the Lingpipe might not give all the result of recognizing for each sentence of input. At that time, the Abner could be able to offer the gene name. Thus, we would choose the processed result of Abner as the current gene tag. This processed method is wrote in this Annotator.

The result of using Abner and Lingpipe, and filtering the more precise data with Confidence Annotator:

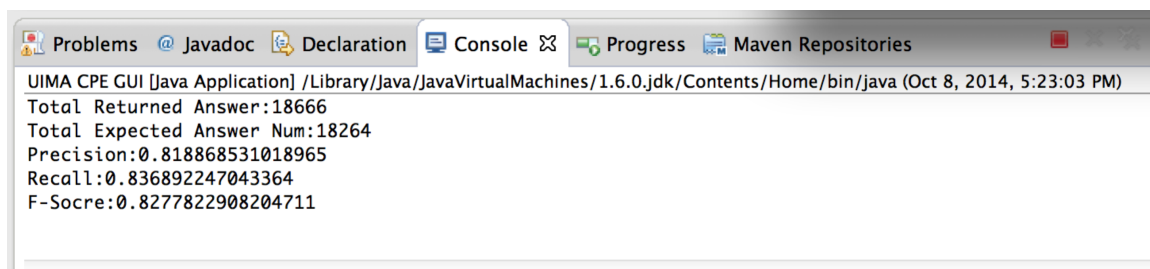


Figure 6. The result of filtering the combination of Lingpipe and Abner.

1.4.1 Algorithm Design

From the testing result of the homework1, the Stanford-NLP has the lowest performance in such a Bio-NER condition. The F-Score of the Stanford just could approximately get 0.3. This final evaluation could not reach the basic demand for extracting the precise gene tag. However, the F-Score by using Abner individually also was around the 0.5, and the F-Score by using the Lingpipe would get 0.75. This

performance is satisfied. 1`1Combining this two API could be lower than 0.8 due to the Abner performance. Thus, to achieve more reasonable output, I wrote another annotator to filter the useful information given by the Lingpipe and Abner. The algorithm in the third annotator (ConfidenceAnnotator) used 2 Hashmap to store the information from the Abner and Lingpipe individually. When the gene tag's confidence from Lingpipe is greater than or equal to 0.6, the possible gene tags will be saved in compared hashmap, and give the information to the Consumer type system. When the gene tag's confidence from Lingpipe is lower than 0.6, we will consider to output the gene tag from Abner results. But, if the Lingpipe could give the same output, we will trust the result from the Lingpipe.

1.5 Design of The CasConsumer

I used the same Casconsumer in the homework 1, I used the iterator to process all the CAS from confidence annotator. In the meantime, I realize that removes all the space and received all the accurate start and end position of gene tag. Finally, I use BufferedWriter method to write all possible gene ID, the start position, the end position, and the gene name tag in the output file.

2. Summary

In this task, the most difficult part for me was to design the flow of the annotators and how to process the raw data as precise as possible. Integrating different annotator is the part that I learned from this homework. With adding another annotator to filter the possible gene nametag, the precision will improve simultaneously. I hope this idea could be utilized in the coming homework.