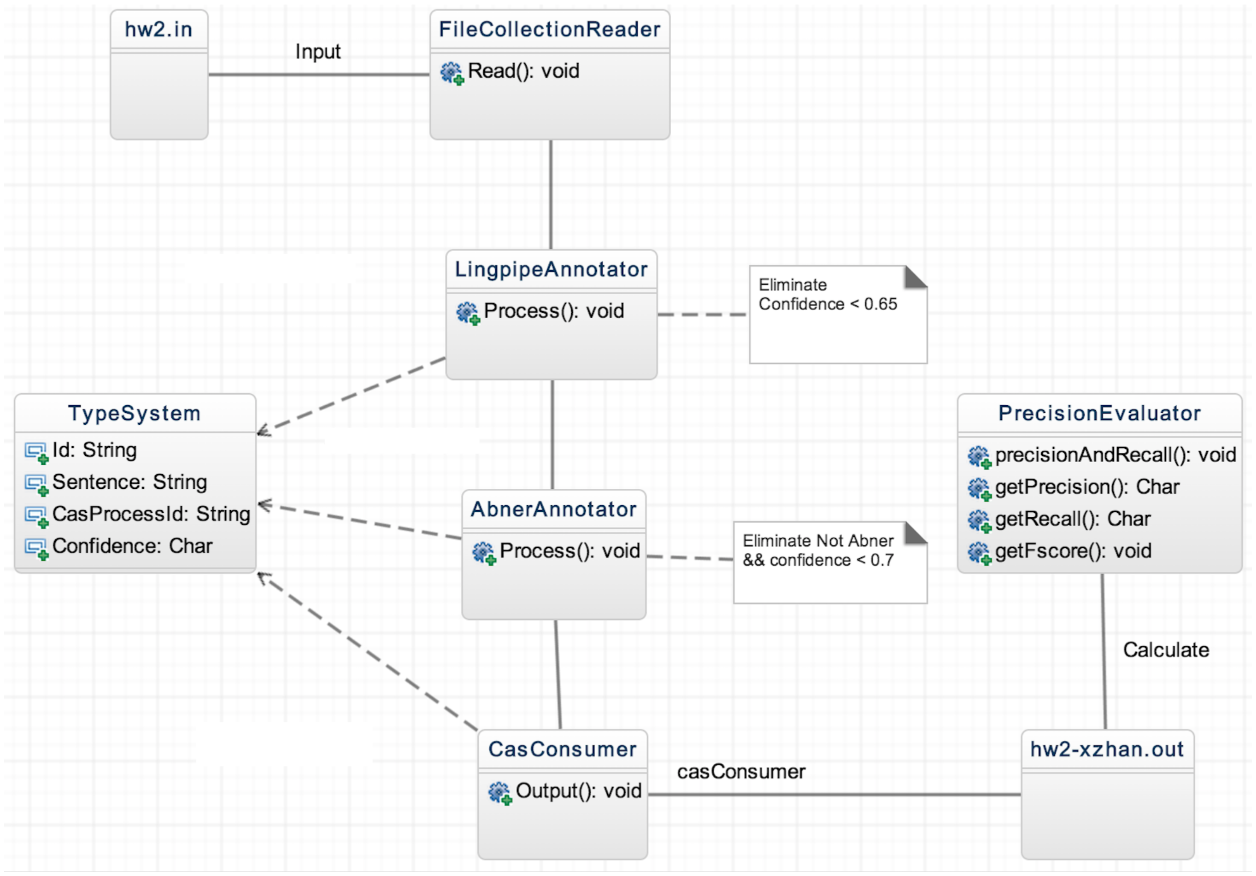
1. Design aspects

To use at least two kind of resources, I first checked the performance of several resources and also asked some of my classmate’s experience. So I decide to choose two best performance resources: LingPipe and Abner.

The architecture and general data flow can be seen below:



1.1 Collection reader (FileCollectionReader):

The collection reader use basic Java I/O to read file. By using the getNext() function, it read the input file line by line, then put the line document into JCas for Analysis Engine usage.

1.2 Type System (type):

To keep my project simple, I only use the Annotation Type, and add Id type and Sentence type

Type extends Annotator with start and end, Annotation type contains CasProcessId for keep which annotator has process the annotation, confidence for the model's confidence towards each annotation.

Also add two more new parameters:
id: store the id of the sentence/geneTag
Sentence: store the original text information

1.3 Analysis Engine:

I combined two Annotator with a aggregate annotator. LingPipeAnnotator and AbnerAnnotator.

1.3.1 LingPipeAnnotator:

I read the info from Jcas and use hmm model to analyze the content line by line, store the line id, line content, the confidence, into the annotation.

1.3.2 AbnerAnnotator:

Use Abner to refine the annotation get from LingPipeAnnotator, remove those are no likely agree by LingPipe and Abner, also remove RNA.

1.4 CAS consumer (GeneCASConsumer):

The CAS consumer read the type from the JCas passed from annotator, then calculate the right start and end position of the Gene Tag in the sentence. Finally write the id, gene name and start and end position in the assigned formation into the output file. In addition, I add a test class which will calculate the precision, recall and F score of my out put. Which will shown in the console line.

2. Algorithm Design

I know we shouldn't use LingPipe for overfitting, So I used another trained model to test and train my system. Thus I can use LingPipe now.

Since Abner doesn't have confidence attribute, when filtrating different result, I appoint Abner as a counterpart.

So I decide to first refine input data by LingPipe, then filtrate the result by Abner.

LingPipe has a method of return first N best result, I choose 5. Anyway, we will eliminate eligible result by measuring confidence of the result, so more return results give me more choices.

Then, I use the Abner to eliminate ineligible results. If the result from LingPipe is not eligible by Abner, I have to check the confidence from LingPipe of the word. If it is below a certain value, definitely should be beyond the value to choose results from LingPipe, then this word should be abandoned.

In addition, I have to use the type recognition function of Abner to eliminate words that are not DNA or Proteins.

At last, output results filtrated by both resources.

So my job is to change the 2 parameters to get a relatively high f1 score.

3. Performance Evaluate:

When only using Abner, the f1 score is only about 0.47. LingPipe is about 0.7

Then I tried several times of different configuration, when 0.6 for the configuration limit of Lingpipe, 0.65 for Abner, I get the result:

Hitting number: 13970

```
Number of sample.out: 18265
Number of hw2-xzhan: 17443
The precise of our system is 0.8008943415696841
The recall of our system is 0.7648508075554339
The f-score of our system is 0.78245771255741
```

4. Analysis

My design has a flaw.

Since unlike other students use both resources to analyze input doc and merge the results of them, I only use Abner to eliminate ineligible result. Thus making my design cannot get high f-score. Because I don’t take into account cases when Abner is right while PipeLing is wrong.

But my design can get the result faster than theirs.

This is a tradeoff. But for this case with small data, maybe my solution is not that good.

Thank you!