HW2 Report

Andrew ID : yichenca

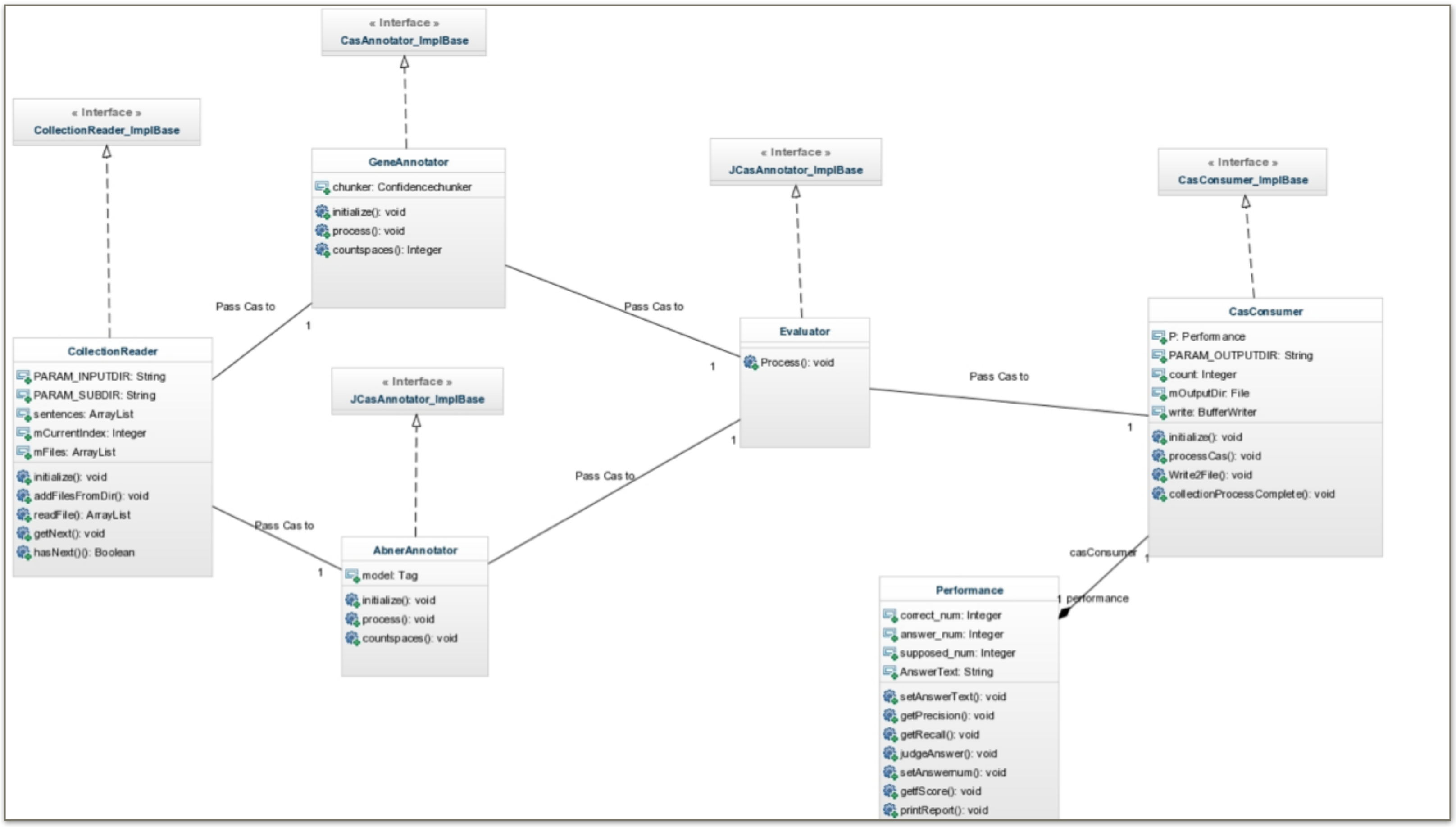
Author: Cai Yichen

Date: 2014-10-07

1. Design Aspect

My system consists of 6 classes primarily, which are CollectionReader, GeneAnnotator, AbnerAnnotator, Evaluator, CasConsumer, Performance. The CollectionReader reads file line by line, and sets annotation in JCas. The GeneAnnotator and AbnerAnnotator will recognize the gene by LingPipe and Abner, the third party library, respectively. Then the Evaluator will collect and judge the results, then pass the final results to CasConsumer.

The class diagram below shows the relationship among these classes.



2. Implementation

2.1 Type System

edu.cmu.deiis.types.Annotation

This is a super type, which consists of casProcessorId and confidence.

edu.cmu.deiis.types.Text

As input file consists of lots of sentences and each sentence includes the only id and a couple of words, therefore I design Type Text. It has a id (type String) to save sentence id, and has a sentence (type String) , to save content of sentence. It inherits from Annotation.

edu.cmu.deiis.types.Genetype

Genetype has a String id since every gene name is belong to a sentence. It also has a String gene to save a gene mention. In addition, features of geneTag like begin and end are used for recording postion of gene term in sentences. It inherits from Annotation.

edu.cmu.deiis.types.Token

Token stores the results after judging and comparing the recognized gene from LingPipe and Abner.

2.2 Collection Process Engine

There are three main components in CPE, Collection Reader, Analysis Engine, CAS Consumer. These three parts are in charge of processing input, analyze input, and output result.

2.2.1 Collection Reader

The CollectionReader reads a sentence line by line from file, and split the sentence into id and sentence text, then put them into Text annotation, respectively.

2.2.2 Aggregate Analysis Engine

There is an aggregate analysis engine, in which, I have three annotators, GeneAnnotator, AbnerAnnotator and Evaluator.

GeneAnnotator gets the feature of Text annotation, and uses LingPipe to recognize specific gene, then put the casProcessorId, confidence, id, start point, end point and gene into Genetype annotation, respectively.

AbnerAnnotator is another method to recognize gene by Abner, a third party library as well. The operation is similar to GeneAnnotator, after analyzing a specific sentence, it will put the casProcessorId, confidence, id, start point, end point and gene into Genetype annotation, respectively.
Evaluator is responsible for comparing the genes recognized from LingPipe and Abner. It will judge the final results by two rules : 1. if the confidence of gene from Lingpipe is bigger than 0.6, then output it. 2. if the confidence of gene from Lingpipe is between 0.35 and 0.6, meanwhile, the Abner also recognize this gene, then output it. I also used Lingpipe to train another model by adding another 5000 training set. Therefore, it will perform better for an unknown testing set.

2.2.3 Cas Consumer

CAS consumer is mainly to open a target file, fetch token type from CAS and write records into target file. In this consumer, I use information in the out result to calculate the performance of my NER based on the results from sample.out.

3. Algorithm Aspect

Originally, I use StanfordNLP as one method to recognize gene. But unfortunately, the performance of StanfordNLP is so bad that precision is about 0.1 and recall is about 0.5. Thus, I discard StanfordNLP, and start to use LingPipe as the main method to recognize gene. For the name entity recognition, the LingPipe runs a statistical named entity Recognizer, specifically for gene, which is “ne-en-bio-genetag.HmmChunker” , the object chunker can load the begin position and end position of recognized gene.

In this task, I use the confidence named entity chunking to return chunks in order of confidence, which I take to be the probability of the chunk given the input text. Actually, I set the max NBest chunks 30. That is Lingpipe will return the top 30 chunks. Moreover, I screen the chunks whose confidence is below 0.35. By using the model ne-en-bio-genetag.HmmChunker provided by LingPipe website, its performance is pretty good, the F-score could reach 0.81. In addition, I find another testing set whose file name is 5000.in to test the performance of LingPipe. The result shows that it is also acceptable. The F-score is about 0.63.

To realize aggregate analysis engine, I use Abner as the second method to recognize gene. The average performance of Abner is lower than LingPipe. The Precision is 0.47; Recall is 0.51; and F-score is 0.49 by testing sample.in. Therefore, I decide to use Abner as the assisting method.

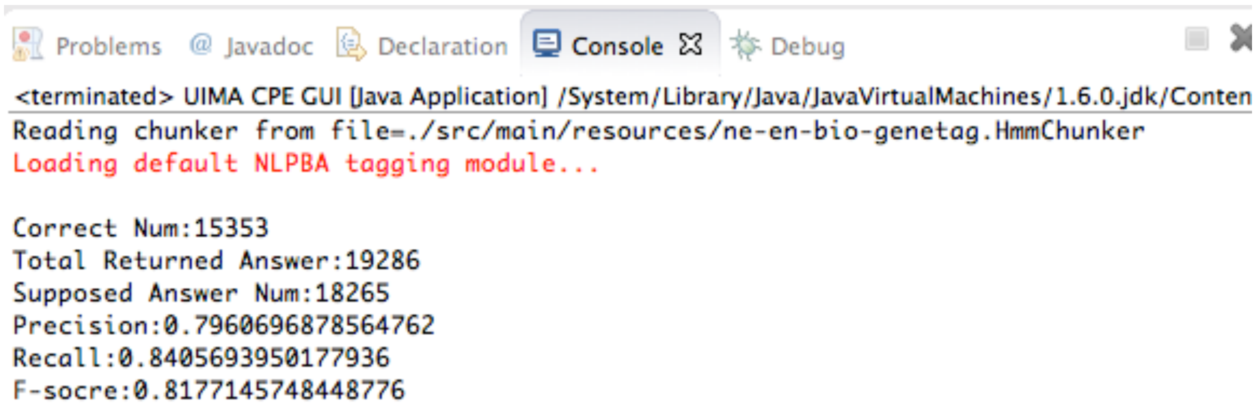
Through testing, I select the strategy as below:
1. if the confidence of recognized gene from Lingpipe is bigger than 0.6, I will consider it’s a correct gene name .
2. if the confidence of recognized gene from Lingpipe is between 0.35 and 0.6, meanwhile, the Abner also recognizes this gene exactly, I will also consider it’s a correct gene name.

By using these two methods, the performance will increase about 3%.

Besides, to improve the performance of LingPipe, I use the sample.in plus another 5000 sentences as training set to train a better statistical model, which is MyModel.

4. Testing

First, I use sample.in as test set, and sample.out as standard result. The performance is below:
Precision is 0.79; Recall is 0.84; F-score is 0.81.

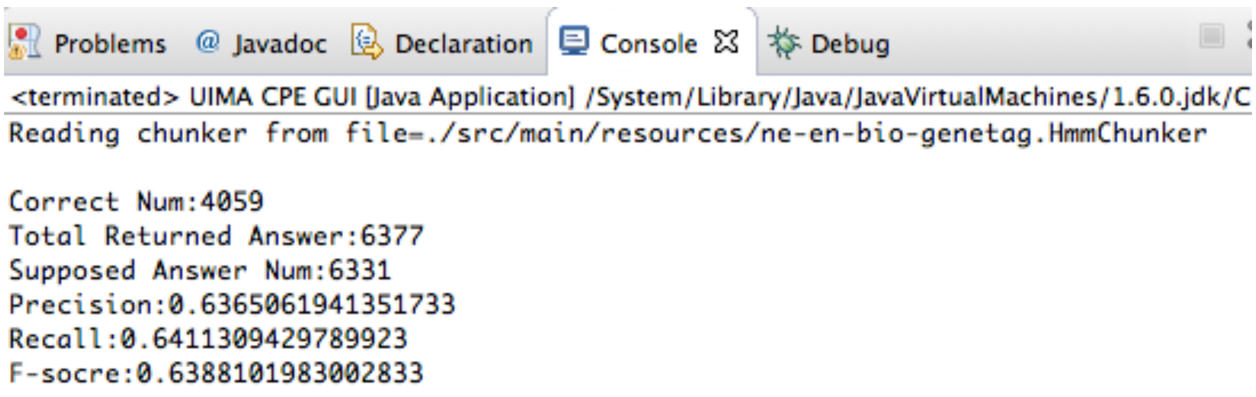


```
<terminated> UIMA CPE GUI [Java Application] /System/Library/Java/JavaVirtualMachines/1.6.0.jdk/Conten
Reading chunker from file=./src/main/resources/ne-en-bio-genetag.HmmChunker
Loading default NLPBA tagging module...

Correct Num:15353
Total Returned Answer:19286
Supposed Answer Num:18265
Precision:0.7960696878564762
Recall:0.8405693950177936
F-socre:0.8177145748448776
```

Then, to resolve over fitting, I use another 5000 testing set which is 5000.in as the test set, and 5000.out as the standard result.

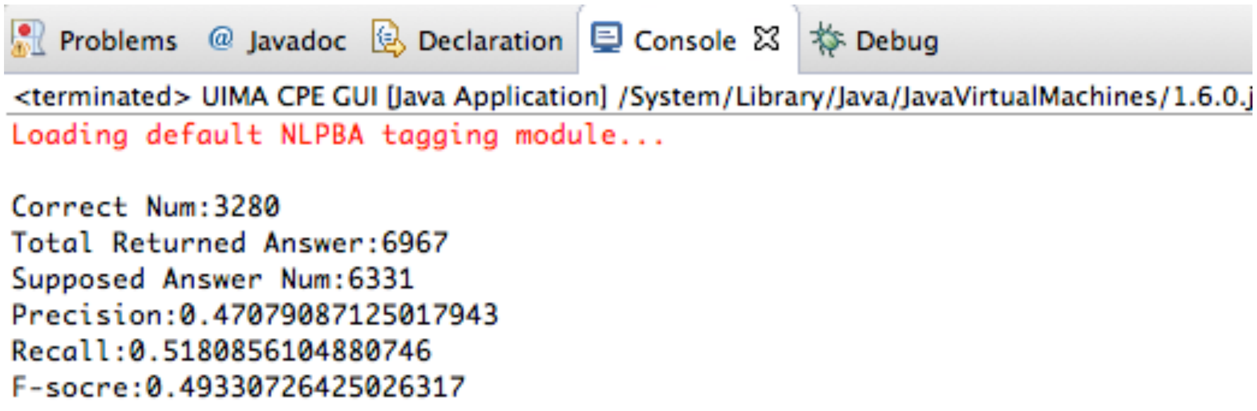
While only using LingPipe, the performance is below :
Precision is 0.63; Recall is 0.64; F-score is 0.64.



```
<terminated> UIMA CPE GUI [Java Application] /System/Library/Java/JavaVirtualMachines/1.6.0.jdk/C
Reading chunker from file=./src/main/resources/ne-en-bio-genetag.HmmChunker

Correct Num:4059
Total Returned Answer:6377
Supposed Answer Num:6331
Precision:0.6365061941351733
Recall:0.6411309429789923
F-socre:0.6388101983002833
```

While only using Abner, the performance is below
Precision is 0.47; Recall is 0.51; F-score is 0.49.

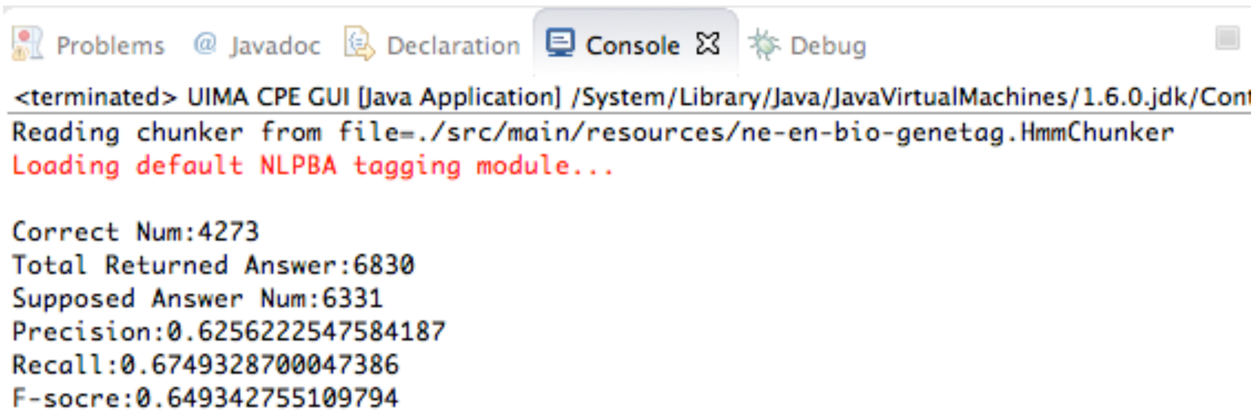


```
<terminated> UIMA CPE GUI [Java Application] /System/Library/Java/JavaVirtualMachines/1.6.0.j
Loading default NLPBA tagging module...

Correct Num:3280
Total Returned Answer:6967
Supposed Answer Num:6331
Precision:0.47079087125017943
Recall:0.5180856104880746
F-socre:0.49330726425026317
```

While using both LingPipe and Abner :

This test shows that the performance of combing both LingPipe and Abner will be better than each of them.
Precision is 0.62; Recall is 0.67; F-score is 0.65.



```
<terminated> UIMA CPE GUI [Java Application] /System/Library/Java/JavaVirtualMachines/1.6.0.jdk/Cont
Reading chunker from file=./src/main/resources/ne-en-bio-genetag.HmmChunker
Loading default NLPBA tagging module...

Correct Num:4273
Total Returned Answer:6830
Supposed Answer Num:6331
Precision:0.6256222547584187
Recall:0.6749328700047386
F-socre:0.649342755109794
```