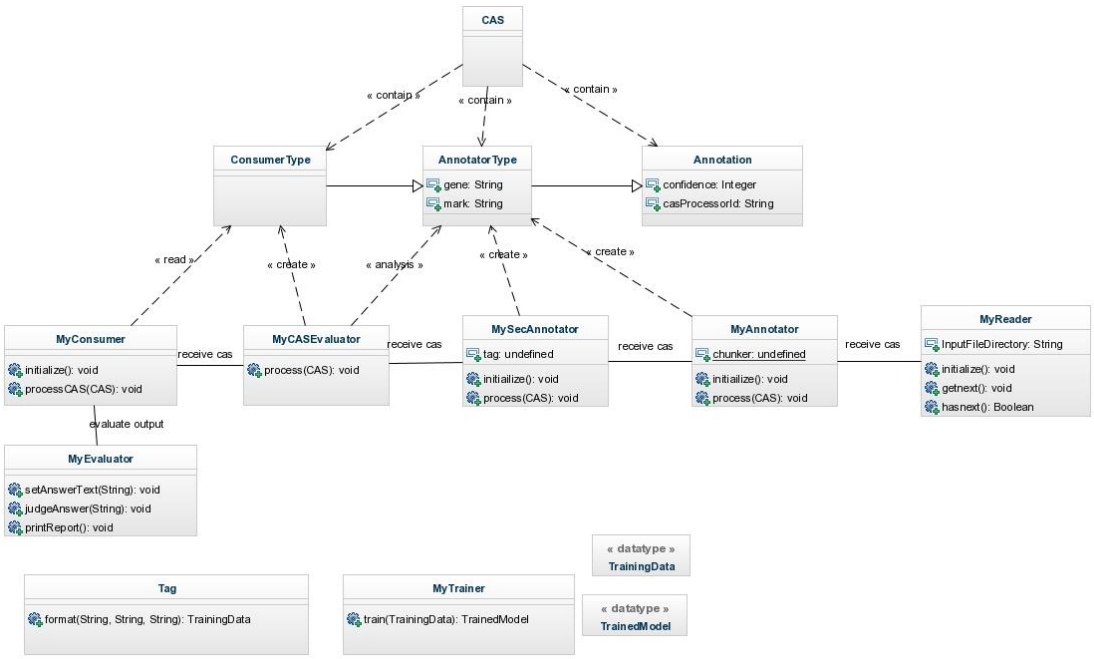
11-693 Software Methods for Biotechnology

Homework 2 report

Xuwei Zou
xuweiz

1. Architecture of system

Class domain diagram:



Structure:

1. 1 TypeSystem:

1) Annotation

Annotation type system is provided by deiis_types. It contains two types: confidence and casProcessorId. Confidence records the degree of one annotator believes a word or phrase is a gene tag, it has a value between 0 and 1.0 . CasProcessorId records which annotator create this type system.

2) AnnotatorType

AnnotatorType inherits Annotation type system. It is mainly used between collection reader and annotators. To fulfill the request of this task, annotatortype contains two new types: gene and mark. Gene stores the word of possible gene word. Mark stores the sentence id.

3) ConsumerType

ConsumerType inherits AnnotatorType type sytem. It is used to distinguish output of reader and LingPipe annotator and ABNER annotator from final result. CAS Consumer will read this type and output data store in this type as final output.

1.2 Collection Reader:

The Reader is responsible for read document, separate ID and text and create CAS for Annotator. For each line in the document, a CAS is created.

1.3 Analysis Engine:

The Engine contains three Annotators: MyAnnotator (LingPipe Annotator), MySecAnnotator (ABNER Annotator), MyCASEvaluator (output analysis annotator).

1) LingPipe Annotator will use LingPipe tool to analyze sentence and sentence id created by the Reader and produce gene tags and offsets. Then the LingPipe Annotator will create new AnnotatorType and store gene name, offsets, sentence id, confidence and casProcessId in the type system.

2) ABNER Annotator uses ABNER tool to analyze sentence and sentence id. ABNER Annotator also uses AnnotatorType to store output. Since ABNER only has no confidence interface. ABNER Annotator will store default 0 confidence in AnotatorType.

3) CASEvaluator annotator will analysis output of the two previous annotators. Then put the result into ConsumerType.

1.4 CAS Consumer:

The Consumer is designed to format the output line and output final result. The Consumer will receive sentence ID, gene tags and offsets in the CAS and calculate the gene tags' offsets (exclude white space) in the formation of given sample.

1.5 Evaluator:

An Evaluator class is created to calculate the performance. But the Evaluator is removed from consumer to release success and avoid running problems.

1.6 LingPipe corpus trainer and Tag:

The trainer retrieve training file in a certain pattern and produce a model used as base corpus pool for LingPipe analysis tool. However due to trainer need an older version LingPipe package, it cannot compile success under 4.1 version LingPipe.jar or upload as .java file. So I upload it as a .txt file under src/main/resources.

Tag is used to convert standard input file and gold answer file into training file.

2. Algorithms and tools used

2.1 Test sample and Corpus training

Since the corpus of LingPipe is created using sample.in and sample.out data. The performance of LingPipe will be overestimated. Thus another source test sample and gold answer are needed. BioCreative website provides 5000 sentences and answers as test sample and 15000 sentences and answers as corpus data. In the performance test, all results are under conditions that use model build on 15000 sentences and use 5000 sentences as test file. New data store in src/main/resources/data/test.in and src/main/resources/data/GENE.eval.

To improve the total performance and avoid over fitting problems. I also train a corpus by using all those 20000 sentences and answers.

2.2 Analysis method

By testing, I find the LingPipe tool has a high performance even when analyze unfamiliar data sets. Also the LingPipe tool provides a confidence parameter to evaluate the possibility of a gene tag is correct. So the ABNER annotator is used as an auxiliary annotator.

In the LingPipe Annotator, the LingPipe tool is used to analyze the gene tag in the sentence, and retrieve words and phrases that LingPipe has greater than 0.4 confidence.
In the ABNER Annotator, all words and phrases are retrieved.
In CASEvaluator Annotator, all outputs from ABNER are stored in a HashMap, and all outputs from Lingpipe are stored in a ListArray. For each results retrieved from LingPipe tool, if the confidence is greater than 0.8 then add it into ConsumerType to output. If the confidence is between 0.8 and 0.6, then check if there are similar output provided by ABNER. If the confidence is between 0.6 and 0.4 then check if there are exact same output in ABNER result.
The result will be processed by a function to exclude obvious not true result, e.g. output with only one bracket.

3. Testing and performance

3.1 Base performance

1) ABNER tool with unfamiliar input
Correct Num:3114
Total Returned Answer:6967
Supposed Answer Num:6331
Precision:0.44696426008324963
Recall:0.4918654241036171
F-socre:0.46834110392540235

2) LingPipe tool output with confidence greater than 0.6 with unfamiliar input
Correct Num:4059
Total Returned Answer:6314
Supposed Answer Num:6331
Precision:0.6428571428571429
Recall:0.6411309429789923
F-socre:0.6419928825622776

It is obvious LingPipe has better performance to analyze an unfamiliar dataset.

3.2 Performance testing

1)LingPipe tool output with confidence greater than 0.8 and output with confidence 0.6-0.8 sharing similarity with ABNER
Correct Num:3886
Total Returned Answer:5663
Supposed Answer Num:6331
Precision:0.6862087232915416
Recall:0.6138050860843469
F-socre:0.6479906619976655

2)Final result
Correct Num:4049

Total Returned Answer:6000
Supposed Answer Num:6331
Precision:0.6748333333333333
Recall:0.6395514136787237
F-socre:0.6567188386992132