HW2- 11-791 Date: 5th October 2014

(HW2) Implementing A Simple Information Processing Task with UIMA SDK

Name: Niloy Gupta AndrewID: niloyg

Pipeline Description:

As depicted in Fig1, the information pipeline is essentially broken down into 5 phases:

1. Collection Reader: Reads the input text line by line and passes it to the Analysis Engine.
2. Aggregate Analysis Engine:
a) GeneAnnotator (LingPipe): This module uses the LingPipe package which uses a previous trained model (model.hmm) to extract named entities. The library uses a Hidden Markov Model N-Best chunker which returns the N best results with confidence estimates.

b) AbnerGeneAnnotator (ABNER): This module uses ABNER Named Entity package which uses the previous trained model "BIOCREATIVE" . This builds upon previous LingPipe annotator. If the gene tags extracted in the previous stage have a confidence value between 30-50% the gene text is passed through ABNER. If ABNER recognizes the entity then the confidence is increased, else it is marginally reduced. The reason so because initial experiments revealed that LingPipe is more accurate compared to ABNER in gene tagging.

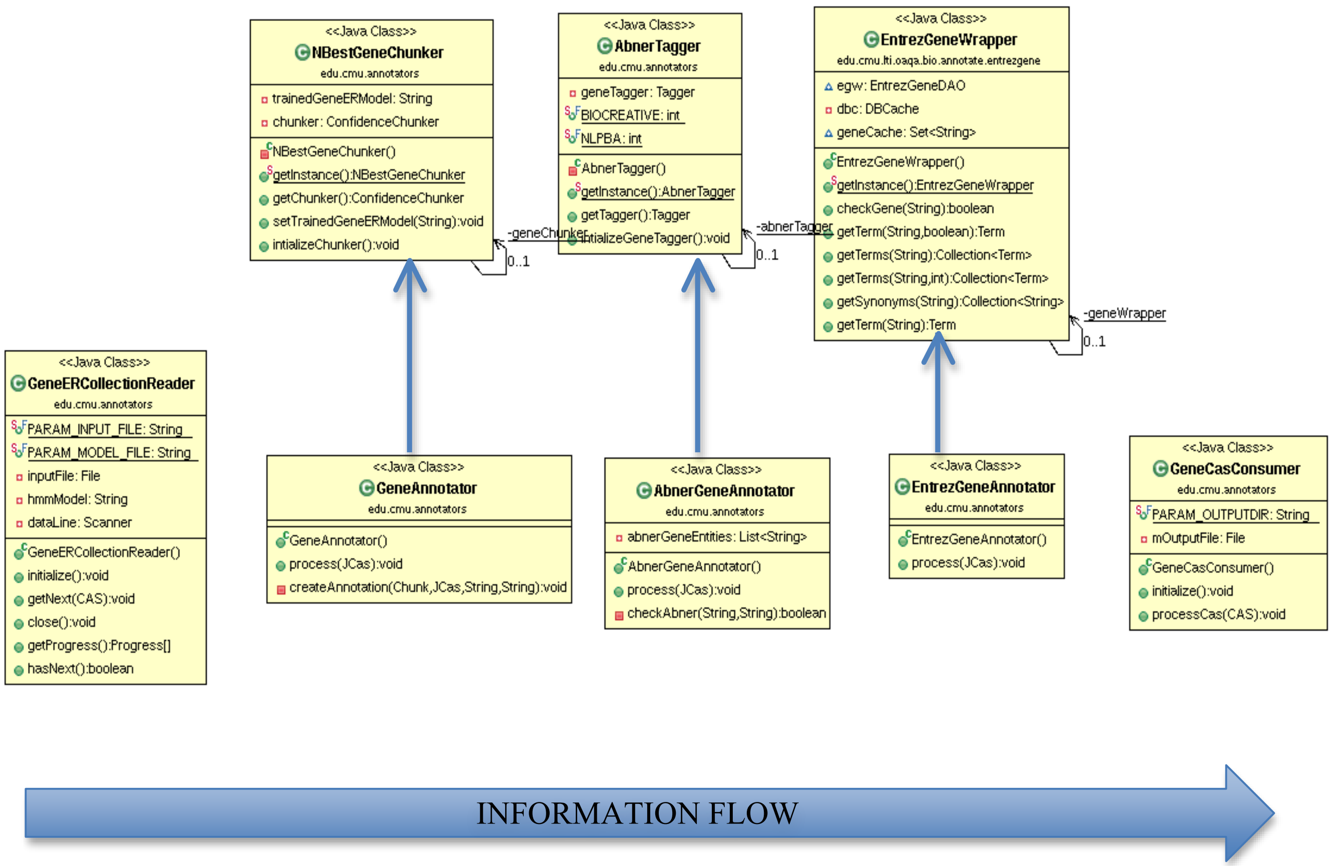
c) EntrezGeneAnnotator*: Unlike the previous annotators which use machine learning methods, this annotator checks for entities by querying the Entrez Gene database hosted by NCBI (National Center for Biotechnology Information) . This annotator queries for only those gene tags that have a lower confidence value. If the entity is available in the database, the confidence is boosted. This was done to ensure minimal web requests were made to the EntrezGene server.

3. CAS Consumer: The CAS after having been populated by the Aggregate Analysis Engine, now has the final confidence value. Only entities with confidence greater than 50% are printed to the file in the specified formart.

Additional Design Notes:
1. The Annotators are initialized in Collection Reader. Each instance of the entity tagger is initialized in a singleton inorder to save the cost of reinitializing and loading the model file for every sentence.

2. The logic of entity extraction is separated from the main annotator class inorder to reduce coupling.

*The EntrezGene wrapper written is a refactored version of the EntrezGene 1.1.2 source code. Methods relevant to the scope of the assignment were preserved. This was done as the older and available version of EntrezGene in the course repository was insufficient for the task.
Link to the original source code:
http://mu.lti.cs.cmu.edu:8081/nexus/content/groups/public/edu/cmu/lti/oaqa/bio/annotate/entrezgene-wrapper/1.1.2/


<<Java Class>>
NBestGeneChunker
edu.cmu.annotators
trainedGeneERModel: String
chunker: ConfidenceChunker
NBestGeneChunker()
getInstance():NBestGeneChunker
getChunker():ConfidenceChunker
setTrainedGeneERModel(String):void
intializeChunker():void
-geneChunker
0..1
<<Java Class>>
AbnerTagger
edu.cmu.annotators
geneTagger: Tagger
BIOCREATIVE: int
NLPBA: int
AbnerTagger()
getInstance():AbnerTagger
getTagger():Tagger
intializeGeneTagger():void
-abnerTagger
0..1
<<Java Class>>
EntrezGeneWrapper
edu.cmu.lti.oaqa.bio.annotate.entrezgene
egw: EntrezGeneDAO
dbc: DBCache
geneCache: Set<String>
EntrezGeneWrapper()
getInstance():EntrezGeneWrapper
checkGene(String):boolean
getTerm(String,boolean):Term
getTerms(String):Collection<Term>
getTerms(String,int):Collection<Term>
getSynonyms(String):Collection<String>
getTerm(String):Term
-geneWrapper
0..1
<<Java Class>>
GeneERCollectionReader
edu.cmu.annotators
PARAM_INPUT_FILE: String
PARAM_MODEL_FILE: String
inputFile: File
hmmModel: String
dataLine: Scanner
GeneERCollectionReader()
initialize():void
getNext(CAS):void
close():void
getProgress():Progress[]
hasNext():boolean
<<Java Class>>
GeneAnnotator
edu.cmu.annotators
GeneAnnotator()
process(JCas):void
createAnnotation(Chunk,JCas,String,String):void
<<Java Class>>
AbnerGeneAnnotator
edu.cmu.annotators
abnerGeneEntities: List<String>
AbnerGeneAnnotator()
process(JCas):void
checkAbner(String,String):boolean
<<Java Class>>
EntrezGeneAnnotator
edu.cmu.annotators
EntrezGeneAnnotator()
process(JCas):void
<<Java Class>>
GeneCasConsumer
edu.cmu.annotators
PARAM_OUTPUTDIR: String
mOutputFile: File
GeneCasConsumer()
initialize():void
processCas(CAS):void
INFORMATION FLOW