GeneEntity Identifier System Report

Name: Shangqing Zhang
AndrewID: shangqiz

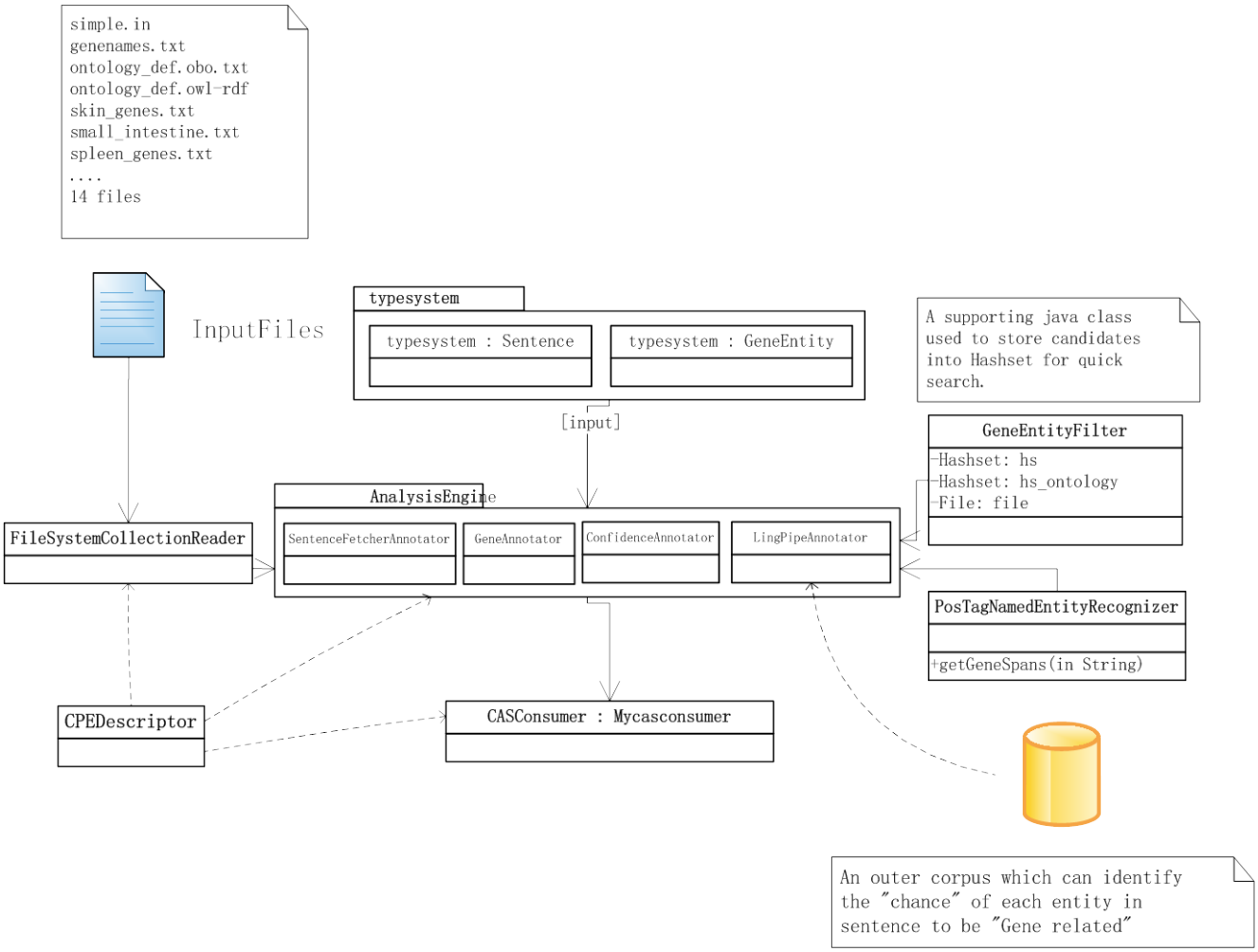


Figure 1. GeneEntityIdentifier System Structure

Figure 1 demonstrates my design at a high level. The system follows the UIMA Framework, using a CPE Descriptor to link 1 Collection Reader, 1 AnalysisEngine and 1 CASConsumer.

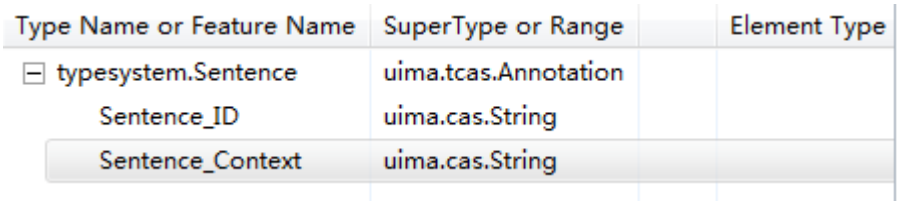
Data Flow:

InputFiles→ FileSystemCollectionReader → AnalysisEngine (Composed by 4 Annotator) → CasConsumer → output.txt (Final Output)

Type System

I use two type systems in this homework

<typesystem.Sentence>



Type Name or Feature Name	SuperType or Range	Element Type
typesystem.Sentence	uima.tcas.Annotation	
Sentence_ID	uima.cas.String	
Sentence_Context	uima.cas.String	

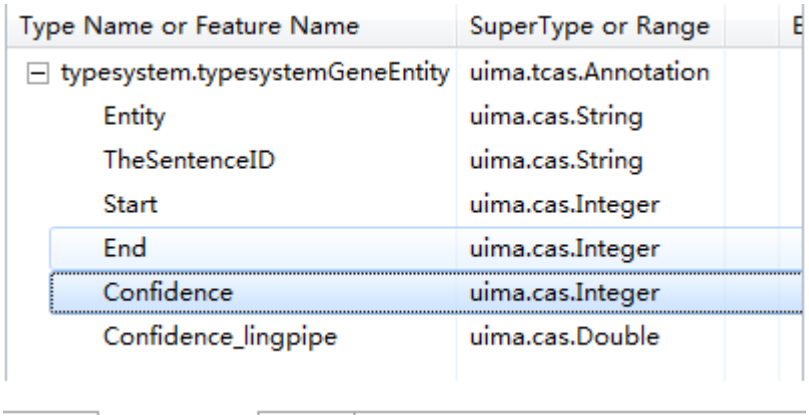
In this first type system, I define Sentence as Annotation, which has two features.

<Sentence_ID> used to store the initial sentenceID in each line of inputfile.

<Sentence Context> used to store the whole sentence in each line.

All the data would be generated after first Annotator, which names SentenceFetcherAnnotator

<typesystem.GeneEntity>



Type Name or Feature Name	SuperType or Range	E
typesystem.typesystemGeneEntity	uima.tcas.Annotation	
Entity	uima.cas.String	
TheSentenceID	uima.cas.String	
Start	uima.cas.Integer	
End	uima.cas.Integer	
Confidence	uima.cas.Integer	
Confidence_lingpipe	uima.cas.Double	

In this second type system, I define typesystemGeneEntity as Annotation, which has five features.

<Entity> This feature is used to store all identified Gene entity from sentence

<TheSentenceID> This feature indicates which sentence the entity belongs to

<Start> Store the starting position of entity

<End> Store the ending position of entity

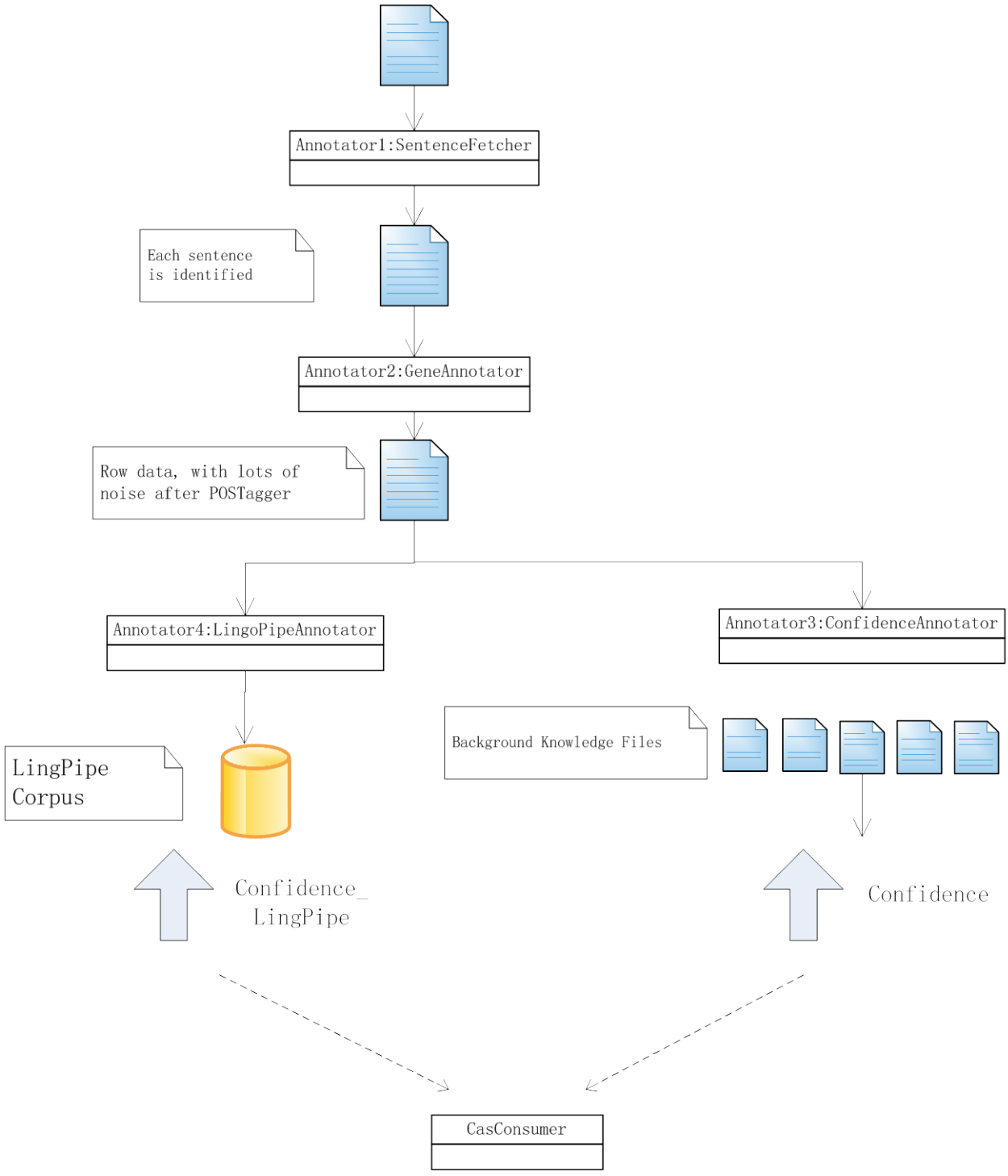
<Confidence>

After filtered by POSTagger, I get a “raw data.” Then this feature is generated after I let this raw data compared with **different files,** which is different from **Confidence_lingpipe**. The confidence would increase by different weight of file. Because some files are more general, some are more specific.

<Confidence_lingpipe>

The reason why I use two different confidence is because they comes from different kinds of analysis. Confidence comes from name_entity files, while Confidence_lingpipe comes from **an outer corpus.** So I use two confidence feature to better identify their roles played in this system.

Annotator Strategy



<First Annotator> : SentenceFetcher
Using SentenceFetcherAnnotator to fetch context by each sentence

<second annotator>: GeneAnnotator
I use iterator to handle each sentence. For each sentence, I first use POSTagger which is given to identify potential “candidates”. Stored in Type System

```
JCas jcas=aJCas;
        FSIterator it = jcas.getAnnotationIndex(Sentence.type).iterator();
        while(it.hasNext()){
      //do the work
}
```

<Third annotator>: LingPipeAnnotator

Using LingPipe to set each “raw data” candidates with **Confidence_Lingpipe**

<Forth annotator>: ConfidenceAnnotator

Using **14 Background files** to set each “raw data” candidates with **Confidence**

<CASConsumer>

Use an iterator to scan **typesystemGeneEntity, can let confidence as a threshold for output**

```
//Combine the two kinds of confidence feature pre-defined to filter the data for output
if(annot.getConfidence_lingpipe()>0.6||annot.getConfidence()>=1){
          fileWriter.write(annot.getTheSentenceID()+"|"+annot.getStart()+"
        "+annot.getEnd()+"|"+annot.getEntity()+"\n");
          fileWriter.flush();
}
```

<external lexical resources (terminology lists) used>

Package:
Location: hw1-shangqiz/lingpipe

Background Files:
genename.txt
ontology_def.obo.txt
ontology_def.owl-rdf.xml
skin_genes.txt
small_intestine_genes.txt
soft_tissue_genes.txt
spleen_genes.txt
stomach_genes.txt
testis_genes.txt
thymus_genes.txt
tongue_genes.txt
uterus_genes.txt
bladder_genes.txt

<Design Patterns>

In this diagram, I try my best to follow the principle of **Low Coupling and High Cohesion**, which means I try to separate each module's role & responsibility as clear as possible.

FileCollectionReader deals with all input files

Each Annotator only handles one particular task:

- Annotator_1 separate sample.in into sentences.
- Annotator_2 generates a row data filtered by POSTagger.
- Annotator_3 set confidence value to each candidate by NAME_ENTITY FILES
- Annotator_3 set confidence value to each candidate by AN OUTER CORPUS

CasConsumer just use the type system and confidence feature to generate output