Report on the UIMA based Gene Mention Tagging

Phani Gadde
pgadde@andrew.cmu.edu

Abstract

This is a brief report on the UIMA pipeline implemented as homework1 for identifying gene mentions in text data. I give a brief overview of the system flow by using the class diagram and the sequence diagram (that are not completely UIMA style) followed by some technical details of mention recognition.

1 System Description

The pipeline was implemented with **two annotators**, IDTextSeparator and NERChunker. A basic CPERunner (provided by the TAs) links the annotators, the CollectionReader and the CasConsumer (NERPrinter).

IDTextSeparator splits each sentence in the input file as ID and Text and updates it in the CAS. NERChunker works on the Text and identifies the entity mentions and adds them to the CAS. The CollectionReader and CasConsumer do standard read and write to the given files respectively. The next few sections give some more details of the pipeline using some UML diagrams.

1.1 Type System

The common type system to all the phases in the pipeline (SharedTypeSystem) contains just two types, **Sentence** and **Gene**. Given below are the details of these types.

```
Sentence
  Features:
    ID: ID for each sentence in the input
    Text: Actual text to process
  Start: Start index of Text
  End: End index of Text

Gene
  Features:
    content: The text of the entity
  Start: Absolute index of the start of the content after removing white spaces
  End: Absolute index of the end of the content after removing white spaces
```

1.2 Class Diagram

Figure 1 below shows the class digram which shows the the associations and the implementation of various phases in the pipeline.

1.3 Sequence Diagram

Figure 2 shows the sequence digram to make the overall data flow clear.

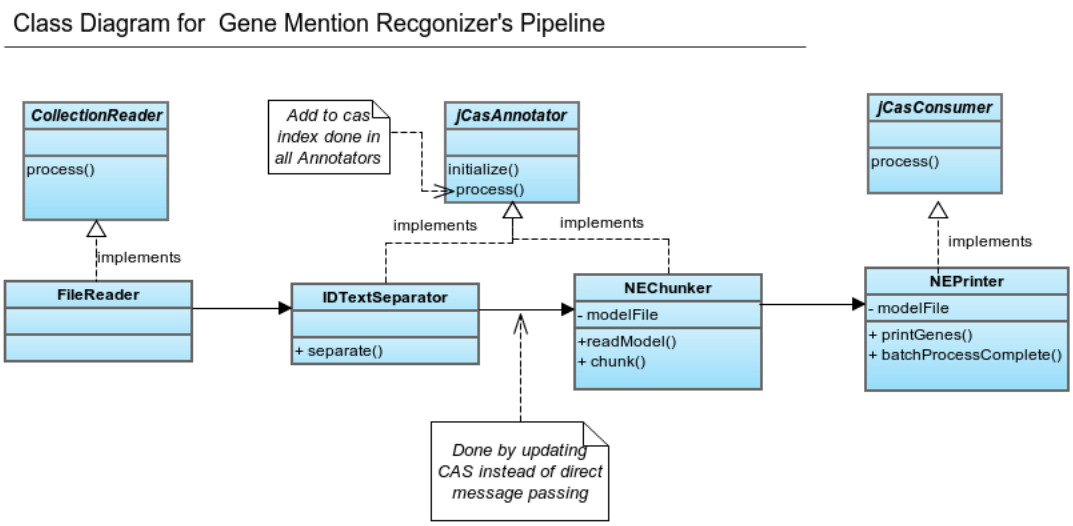

Figure 1: Class diagram for the pipeline

2 Techniques Used

2.1 Lingpipe model

The model used in the NERChunker module is from lingpipe[1]. It is statistical model training on the GENETAG corpus using Hidden Markov Models (explained in the next sections). No other lexicon or rules are used in the entity recognition except this model, which is packaged with the code.

2.2 Machine Learning Techniques

The lingpipe model uses Hidden Markov Models (HMM) to do the entity recognition. HMMs are one of the simplest and effective models for sequence tagging where an entire sequence of classification decisions are optimized together. Tagging a particular word takes into account the tags already given to the previous words which works well in all the tasks that follow such semantics. Named entity tagging is one such task, where the tag for a word depends on the tag of the previous few words. Since the task is effectively handled with less parameters, HMMs are one of the frequently used models especially when the training data is small and the models need to work in real time.

2.3 NLP Techniques

Entity recognition using HMMs is done in the lingpipe model in three steps. First, a given text is tokenized into words. Tokenized text is then passed to the HMM trained sequence tagging model. A standard way in NLP to do chunking as a sequence tagging task is to give B, I and E tags to words meaning Beginning, Intermediate and End respectively. Word sequence within B and E are considered an entity.

2.4 Biological data

The lingpipe was trained on the GENETAG corpus released under creative commons license in 2005. The description and more details about the corpus can be found at http://www.biomedcentral.com/1471-2105/6/S1/S3

[1] http://alias-i.com/lingpipe/

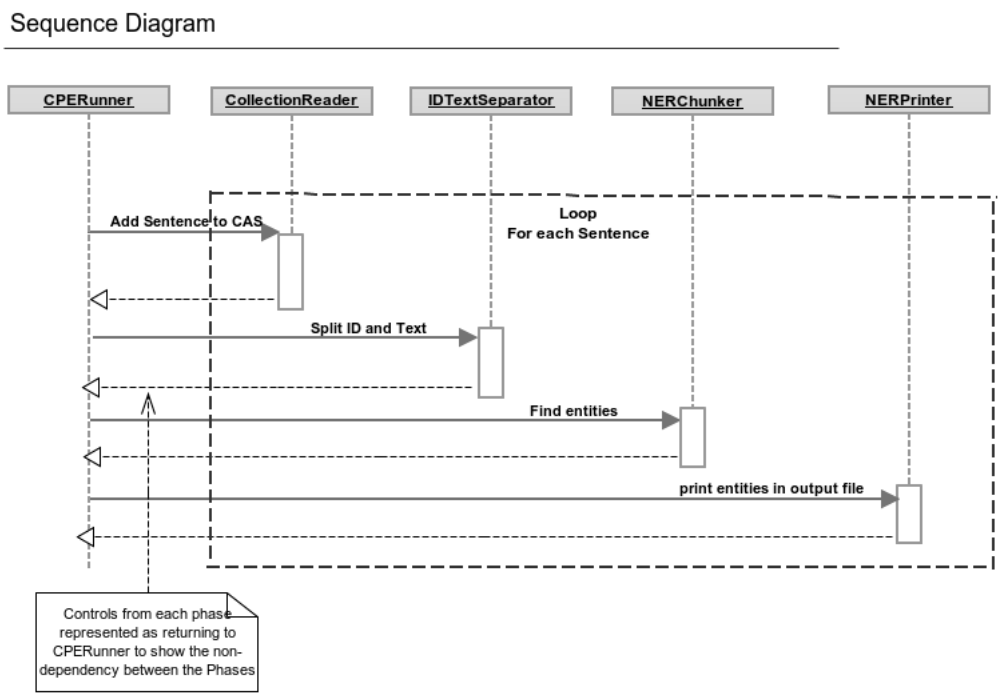

Figure 2: Sequence diagram for the pipeline

3 Evaluation

A basic evaluation was done on the output by this system considering the sample output as the ground-truth for entities on the sample input. While the number of entities in the ground-truth were 18265, this system gave 20174 mentions. However, most of the mentions in the ground-truth were in the output, which denotes that this is a high recall low precision system. The F-measure of the system was above 80.