Design and Engineering of Intelligent Information Systems

Alok Kothari

October 12, 2014

1 Task

The tasks consists of being able to using multiple analysis engine for NER Gene Tagging.

2 Overview

I used two analysis engines (Annotators)

- Hidden Markov Models- based named entity recognizer (lingpipe) for this task. The lingpipe used a model file "English Genes: GeneTag".
- ABNER a CRF-based biomedical NER.

I went through the type system given to me and added the SentenceAnnotation Class - that we had done in the first assignment. The types in it were Sentence and Sentence ID. I also decided to utilize the 'Token Class' – which already extends the Annotation Class. I added "Named Entity String" (nerstring) as an additional type to store the Token Class. The begin, end and other attributes were already inherited from Token.

3 General data flow

The Collection Reader (*NewCollectionReader.java*) reads the input file. The name of the input file is supplied as a parameter: "InputString". It uploads each sentence as a CAS.

This then read by the Annotators, which then annotates the named-entities in the sentence. The two annotators are *MyJCasAnnotator* and

MySecondJCasAnnotator. There are multiple ways to combine output from the the two ways (Details in Experiments Section). Both the annotators use the Token Class to add to JCAS.

The output of both annotators is sent to CAS Consumer. Where I store all the named entities in different HashSet's/HashMap's according to which annotator it came from. I could do this because earlier I set the 'casProcessorID' before pushing the NER's to CAS. Then using these HashMap's I experimented different ways to combine output. I used the very useful **collectionProcessComplete** function. This function is called after *processCas* is done running. Thus it provides a good way to implement various ways to combine outputs from the annotators. The final output is written to file in desired format. The name of the output file is supplied through the parameter 'outputfilename'.

4 Experiments

I will document some of the experiments I did in combining the outputs.

Approach	Prec	Recall	F
Approach 1	0.8517	0.4580	0.5957
Approach 2	0.8046	0.6120	0.6952
Approach 3	0.7787	0.7920	0.7853
Approach 4	0.7638	0.3800	0.5075
Approach 5	0.6335	0.8650	0.7314

A brief explanation about the Approaches.

- Approach 1 Considering only named entities that are common to both annotators.
- Approach 2: Considering only named entities that are common-substrings of each other. This is because Abner sometimes tags more of the string than just the Named Entity Part. Thus a more effective way to combine and consider entities from both annotators is if one is substring of the other.
- Approach 3: Following 2, we can consider only those sentences where both the annotators tag 'some entity'. But since we know lingpipe is more accurate, we just consider output from Lingpipe (if that ID was

tagged by both annotators). However I don't know whether this is 'cheating'. So I include results here for reference sake.

- Approach 4: Do what is done in Approach 2, only this time, consider only those named entities from Abner which are of size more than 6.
- Approach 5: Approach 4, Approach 2 hit recall a lot. So we consider entities from both like in Approach 1, only filtering for size 6 from ABNER annotator. This is the SUBMITTED code.