Working notes

jingmao you

2020-04-21

1 Whe storage of information

All the information is extracted from jCas.
After the KNOWNER pipeline, there will be named entities as the output.

2 What to do next

The goal is to insert an jCas object, in order to run the aida analysis engine. This object should contain the same type of output as KNOWNER pipeline. Build a new pipeline, containing only Aida pipeline. The input should be named enetities, which contains all the information that we need.

3 How to build named entities?

In order to see the named entities, I take a lood at the KnowNERIntegrationTest: KnowNERIntegrationTest
From the testing file, it builds the document first, then it pass it to document-processor. Document processor will process the document, and build the jCas instance. The named entites could be extracted from the jCas.

4 The document processor

- create jCas using JCasFactory
- addSettings to jCas
- set the document into jCas
- process jCas

5 The related links

- JCas
- DocumentProcessor
- KNOWNER
- Disambiguation testing code

6 simulate the result of KNOWNER

Trying to find a way to insert the result of KNOWNER into a jcas instance. This jcas should be able to pass the test method. The idea here is to add the annotations to the original document and the results could be extracted from those annotations.

6.1 The test

The test that I would like to pass is KnowNERIntegrationTest.java. The original method is using document processor to create the jcas instance.
What I have done is using the JCasFactory to create a jcas instance first: JCas ajCas = JCasFactory.createJCas();
Then adding the sentence annotation to the jcas instance: Sentence sent = new Sentence(ajCas, start, start+text.length());
In each sentence annotation, it contains some named entity annotation, so I also annotate some named entities to the jcas instance: NamedEntity ne1 = new NamedEntity(ajCas, start, start+text1.length());
Adding the FeatureStructure to the jcas indexes: ne1.addToIndexes();
Set to value to named entity to pass the test.
Extract the results from the manually built jcas instance: NamedEntities namedEntities = NamedEntities.getNamedEntitiesFromJCas(ajCas);
The running result shows that it could pass the test.

7 The building of the new pipeline

The idea is to modify the input and rebuild the jCas instance when we have the input, and then add the aida anlysis step next to it.

7.1 Classes related to steps

SimpleStep.java: Indicates that a CAS should be routed to a single AnalysisEngine. After it has been processed, the Flow.next() method will be called again to determine the next destination for the CAS.