Final Report

Name: Yancheng Liu

Andrew ID: yanchenl

Date: 09/24/2014

I only use the given Stanford CoreNLP recognizer to finish the task for the reason that I am a biology major student and I really need some time to learn programming. It works but the result is not precise. I am taking the programming class (02-201) this semester so I believe the situation will get better next time.

Accordingly, I choose to revise the RoomNumberAnnotator and XmiWriterCasConsumer given in the UIMA SDK's tutorial and just deployed the existing Stanford CoreNLP recognizer in handout. For the collection reader, I choose to use the FileSystemCollectionReader given in the example, the disadvantage is that it read the whole file directly instead of read line by line, so that the output is sorted in ascending of begin position and descending of the end position. **It looks like the first several line of the output file are unmorally long but it is the right result produced by this given recognizer. (Same line existed in sample.out). And for the reason that this collection reader is read a whole folder, I use /hw1-yanchenl/src/main/resources/data as the input folder. This may different from the requirement which read the hw1.in file.**

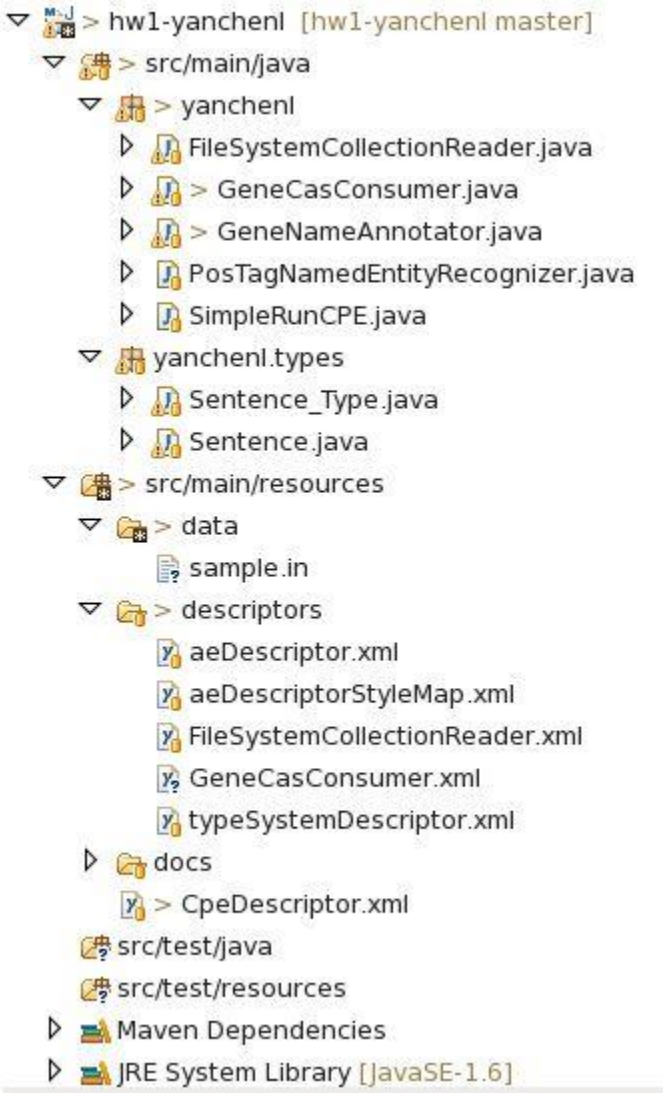
I produced the CPE descriptor by using CPE GUI. The type system Sentence includes two types, which are id and text. Id is used to store the sentence identifier of each sentence and text is used to store the gene and gene product mentioned in the sentence. I created a type system descriptor xml file typeSystemDescriptor.xml to create the type system and the other descriptor file imported it.

For the mechanical of the recognizer, as it says in the homework tutorial, this named entity recognizer tokenizes the sentence into tokens by using a tokenizer. Next, it label each token with a part-of-speech tag. Then, nouns will be grouped and add to the list of named entities.

I didn't use any machine learning technique or external database, although I spent more a lot of hours getting familiar with them and still cannot figure out a way to use them. Javadoc is generated in /hw1-yanchenl/doc.

The detail is listed in the next pages.

Structure are listed here:



Collection reader: FileSystemCollectionReader (from uimaj-example)

Annotator: GeneNameAnnotator (revised from RoomNumberAnnotator in uimaj-example)

Cas consumer: GeneCasConsumer (revised from XmiWriterCasConsumer in uimaj-example)

CPE descriptor: CpeDescriptor.xml produced by CPE GUI. The output directory is the root directory required by the handout and the file is named hw1-yanchenl.out.

Collection Eeader Descriptor: FileSystemCollectionReader.xml

Analysis Engine Descriptor: aeDescriptor.xml

Cas consumer descriptor: GeneCasConsumer.xml

Just to make up the points lost in the algorithm part, I want to try one of the additional points part. I tried to investigate the problem from a biomedical (instead of language technology) view because I am a biology major student.

Most gene names are ordered letters and numbers. Some of them is meaningless for the quick need to assign an identifier. Others carry some information like YAL042W. Y represents the specie is yeast, A represents the chromosome, L indicate the chromosome's left arm, and W is the Watson strand, the number 042 is a identifier. Gene named in the early days are often several letters followed by numbers. This kind of gene name is usually an abbreviation for information describing the findings of the gene[1].

The guidelines for human gene nomenclature was published by Human Gene Nomenclature Committee (HGNC) in 1979 and it was revised many times after that[2]. All the approved gene symbols of home spice are in the Genew database[3]. So it could be an official database for researchers to refer.

For the task of finding gene names, I believe the word "name" here has two meanings which are gene symbols and gene names. Biologists use both during their writing. And one more challenging part is that some genes have many aliases. For example, the famous p53 gene, also known as tumor protein p53, BCC7, LFS1, TRP53, Cellular Tumor Antigen P53, Antigen NY-CO-13, Tumor Suppressor P53 and so forth[4]. You could find that some of them contain the word p53 but others don't.

Genes identified from sequence information including genes from antisense, opposite strand, untranslated functional RNAs, related sequences, pseudogenes, and genes of unknown function. All of them have unique nomenclature[5]. This part also account for a large part of the gene. The rule should be considered during the design of the machine learning method and algothrim.

The task of finding gene and gene products in the sentence is much more challenging than several years ago for the changing naming method and the more unstructured name. I believe refer to the official raw database perhaps the best way to do the task.

Work Cited

1. Seringhaus, Michael R., Philip D. Cayting, and Mark B. Gerstein. "Uncovering trends in gene naming." Genome biology 9.1 (2008): 401.
2. Evans, H. J., J. L. Hamerton, and H. P. Klinger. "Human Gene Mapping 5 (1979): Fifth International Workshop on human gene mapping." Cytogenet Cell Genet 25: 1-4.
3. Wain, Hester M., et al. "Genew: the human gene nomenclature database." Nucleic Acids Research 30.1 (2002): 169-171.
4. Hsieh, Jui-Cheng, et al. "Identification of two novel functional p53 responsive elements in the herpes simplex virus-1 genome." Virology 460 (2014): 45-54.
5. Wojcik, F. "[Guidelines for human gene nomenclature]." Annales de biologie clinique. Vol. 60. No. 3. 2001.