Kenton Murray (kwmurray)

I tried a few different approaches to getting the project up and running. From an NLP perspective, the first AE that I built looked at a list of approximately 9,000 common genes and protein names and tried to identify n-grams of various lengths to see if any matched. Like most things in NLP where lists are concerned, it broke down very easily. It was hurting my overall precision and recall in my system and Out of Vocabulary (OOV) words are a big problem on any test set I looked at. From there, I decided to go with the state-of-the-art ML package LingPipe. Using the HMM NER model that they have implemented, I was able to get a vast improvement in my results. In the final submission, the CPE doesn't include the first method because it was hurting my overall results on a test set. One nice thing about UIMA was how easy it was to swap in different AEs. When one didn't work as well as I had liked, I simply didn't use it. I still had the CASes and Types that I had defined from before and they were consistent across the system, so it didn't matter.

There were two types I used. When reading in the file using the Collection Reader, I had a type that stored the String ID and the String. These were simply stings. When I was identifying genes and proteins, I had a type that stored the String ID, gene, and start and end values.

My hierarchy is not very flat for my source and that is because I anticipated adding to this. The package/directory structure is a little more complicated than needed for this size project, but if I wanted to swap out different methods, it is easier to keep track. For instance, I wrote two different Analysis Engines - though only ended up using one in the final system. I had a separate package for all the Analysis Engines so that it is easier to keep track of. Additionally, I had a separate package for all the CAS Consumers. I only wrote one which prints out in the format specified for this assignment, but if at a later date I wanted to print them

out differently, I wanted these in a separate area. I felt that this design was much more modular.

The flow of my system follows the general UIMA framework. I have a single Collection Reader that reads in a single file from a parameter. It creates CASes, one per line in the original document. This then is passed to one or more Analysis Engines. These annotate a Gene Type. The final system performs better with only one of the AEs, so the CPE Descriptor has changed to allow this. In general, it is modular enough to not matter. Finally, there is one CAS Consumer, which takes a CAS and writes to a file (also specified by a parameter). One nice thing about the design is that it doesn't matter when a Gene is identified, it will be written immediately. As soon as it is indexed in the process method of an AE, the Consumer can run its process method and write to a file. This is good for large systems.