Tyler Lanni

CompLing Methods II

HW 2 Write-Up

Given the skeleton of a program that reads a .tok file, we were asked to finish it by completing several different tasks. The original function provided used a generator to read each line in a for-loop and then return an Iterator[List[List[str]]]. My limited experience with generators gave me a bit of trouble initially, but then the instructions said to simply store that return value in a variable 'corpus' as a list. The format made more sense afterward, as it was just a list of lists.

The assignment seemed straightforward after reading the instructions, but I did have problems with a couple steps. The goal of the assignment was to 1) take an input .tok file and store it in a list variable, 2) randomize each sentence location within the list, 3) place 80% of the content in a 'train' variable, 10% in a 'test' variable, and 10% in a 'dev' variable, 4) write the info in each of those variables to their respective .tok files and make sure the info is formatted similarly to the original input. This program was also required to be command-line run.

My initial problem concerned the structure of the program. My 'main' function was not organized, and I had trouble staying DRY. I initially tried splitting up the info within 'main' via a for-loop and a random number generator. For item in 'corpus', if the number generator (0<n<1) was in range of 0<n<.8, it went into train, if the number was .8<n<.9 it was put into 'dev', and .9<n<1 would be put into 'test'. This looked sloppy and I am quite certain I was missing lines. What I did instead (after taking the instructions left by Kyle quite literally) was create a function that took the data and split it into different variables in place. It was at this time that I realized the randomness had to be seeded as well, so the random number generator I created could not work. This required that I shuffle the list of sentences within 'corpus' based on a pseudorandom seed provided by a console argument. After the shuffle, I put the initial 80% of lines in 'train', and split the remaining 20% in half into 'dev' and 'test'. I then returned those variables to 'main' function.

From there, I sent each variable into a 'write_tags' function. This function was recommended by Kyle to keep things DRY. It's a basic function that takes a list of token sentences and writes them line by line in a similar format to the original in their respective .tok files (train, dev, test).

The most difficult part of this assignment was the organization. Once I found that the instructions limited the number of ways you could create the program, it became easier. IE, the instructions about shuffling based on a seed left me with limited options, but it seemed the best way to do it. Essentially, I wrote out the instructions in my ‘main’ function and figured out how to do each step in separate functions. Each step was then completed in a different function to keep things DRY.