Comp Ling II - Homework 2 - Write-up

Command-line tool to split corpus into groups

Vikram D'Mello

Mar 7, 2021

Assignment Goals

The assignment's goal was clear. I recognized the confluence of the handful of general-purpose command line tools we've been introduced to so far, and value of integrating these techniques in a language-related setting to understand them better. However, I wondered about the real-world NLP task which the assignment intended to build up to. It might have been meaningful to anchor this intermediate step in terms of the eventual language processing goal (most likely a machine learning model of some kind).

Instructions & Guidance

I appreciated the organized presentation of each building block, its purpose, technical considerations, and - most important - detailed guidance in doing it. In particular, the central code block of reading & parsing the source corpus was invaluable.

In particular, the `read_tags` code block had a few foreign concepts for me, which I had not learned either in a prior class or via self-learning: *generator functions* and `yield()` . Were we left to our own devices, I might have tried `csv_reader` with a `read_line for` loop...but I imagine there are performance considerations. Need to explore generator functions further - I found this bit the most rewarding of the whole exercise.

An unexpected uncertainty I faced had nothing to do with the coding itself; rather, I found it ambiguous as to whether we were to split the corpus by word (tagged token) or by sentence. The guidance note "*your exact lengths may differ slightly (because the sentence lengths differ slightly*" added to the uncertainty since I couldn't follow what the comparison referred to...ie, difference between what and what. At any rate, after conferring with classmates, I gather no one else seemed to have this doubt - sentence was apparently clear. So perhaps I missed some obvious specification. In my mind the exercise could have gone either way, depending on the (unknown) downstream language task.

Coding Strategy & Logic

The core programming was easy enough. I enjoyed the fun challenge of coming up with a loop to write each split to its file. I ended up doing so not through a user-defined function - I suppose resulting in a bit of *spaghetti code*. Am unsure if the dictionary approach I came up with is best. Am generally weak and hesitant with UDFs - need to practice more and become more confident with them.

With the randomization operation...beyond the suggested `random.shuffle` technique (which I imagine is the best approach), I also briefly toyed with a completely different strategy, which in hindsight is hare-brained. What if we assigned a random number from 1 to 3 to each & every token to assign it to one of three split groups. We would then keep a tally of each split, stopping when it reached 80% of the source list size. Senseless I guess because it's a ridiculous amount of computation and waste of resources. But I wonder if the approach has use in other contexts or tasks....

Execution & Testing

Easily the single-biggest hurdle of the entire homework was the CLI. The `argparse` coding itself was simple enough, thanks to good class explanation and excellent practicum exercises. However, the ensuing battles with Python versions, Conda installations, and the like were wholly deflating and took far more time and effort than I would have liked. It's still not quite fixed on my computer; however, since it does work perfectly with `python3` explicitly prefixed at the command line, I believe that should satisfy the requirement to work on another machine which is set up correctly.

More important, I found the CLI portion of the assignment the least rewarding. For now I am not seeing the value of creating such tools, since I imagine everyday NLP work occurs chiefly or even exclusively in an IDE rather than from the command line. That said, I'm sure there are some learning goals which we will build towards later.

Lessons Learned

Need to keep an open mind as to what might or might not be useful for now.