Natural Language Processing

Assignment 2

Liangqun Lu

U00624020

**Problem 1.** Write a Perl script that creates a hash of hashes containing words, their parts-of-speech tags (POSes), and frequencies of the tags for the corresponding words from an expert-annotated text file. Assume the words in the file have been manually annotated by experts with POS tags. Keys for the first level of hash should be the words while the values are hashes. The second level of hashing uses the POSes as keys and the values are the frequencies of the POSes for the corresponding words.

(i) [10 points] Write a Perl script that maps each parse tree in the SnapshotBROWN.pos.all.txt file (see the website) into one-line sentences as shown below. You should retain only the parts-of-speech and the words from the parse trees. Each sentence should span a single line in the output file.

Example Output

DT The NNP Fulton NNP County NNP Grand NNP Jury VBD said NNP Friday DT

an NN investigation ... rest of the sentence here

Run the script on the file SnapshotBROWN.pos.all.txt and save the

result in BROWN-clean.pos.txt

(ii) [10 points] Generate the hash of hashes from the clean file BROWN-clean.pos.txt . (iii) [10 points] In BROWN-clean.pos.txt detect the 20 most frequent tags. Report their frequency.

(iv) [10 points] take the most frequent tag and use it to tag the words in all the sentences from the BROWN-clean.pos.txt file. Report the performance of this tagger. See the slides for details on how to measure the performance.

**Code:**

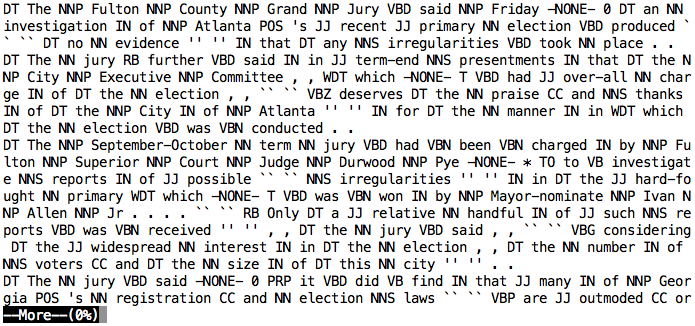
‘**hw2.py**’

**Approach:**

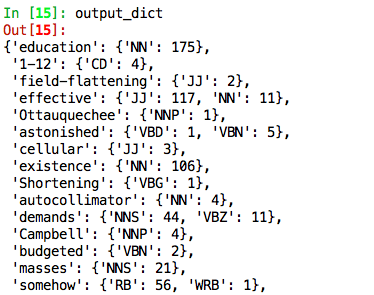
1. Read the file “BROWN.pos.all” and change lines into a long string
2. Split with the sentence “TOP END\_OF\_TEXT\_UNIT” and get a list of content
3. Remove empty element in the list
4. Get the POS and word from each element in the content list
5. Save the POS word into file "BROWN-clean.pos.txt" ###sub problem I
6. Read the file "BROWN-clean.pos.txt"
7. The POS is in even position while word in the odd position
8. Create a dict[word] = {pos:frq} ###sub problem II
9. Get all POSes and count the top 20 most frequent pos ###sub problem III
10. Get the most frequent tag for each word and retag the word using this tag
11. Calculate the accuracy ###sub problem IV

**Results:**

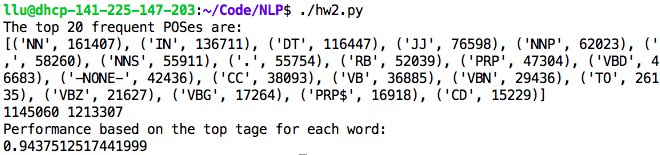
I: A portion of the clean file is like below:



II: hash of hashes from the clean file



III: The top 20 most frequent tags and their frequency

****

IV: The performance is 0.9438 if each word is tagged using the most frequent tag. We retagged the whole content from beginning to end and did not consider the word replication. And the accuracy was calculated overall.