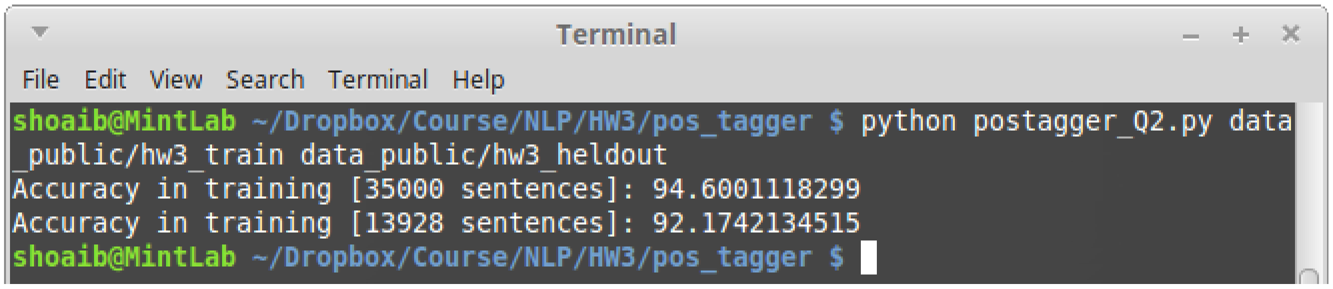
Home work 3 (NLP)
Shoaib Haque Khan
Date: 10/12/2016
--

Q1:

1. Atlanta/NN	Should be NNP	(Proper Noun)
2. Dinner/NNS	Should be NN	(Noun, Singular)
3. Have/VB	Should be VBP	(Verb, non 3rd person singular present)
4. Can/VBP	Should be MD	(Modal)

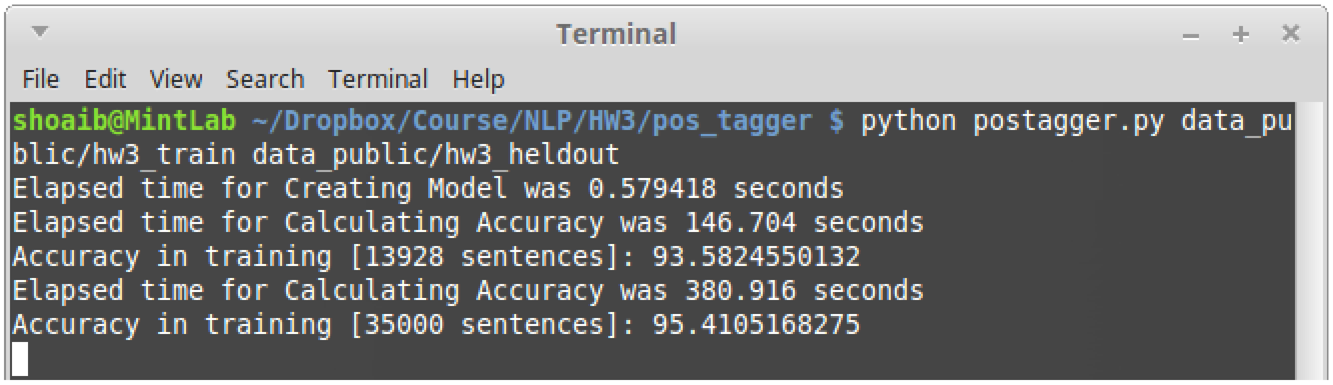
Q2 and Q3:

Findings and results:
For Q2, the accuracy is 94.60% and 92.17% on training data and test data respectively.
It takes around 0.35 seconds to create the model for this question on my machine.



```
shoaib@MintLab ~/Dropbox/Course/NLP/HW3/pos_tagger $ python postagger_Q2.py data
_public/hw3_train data_public/hw3_heldout
Accuracy in training [35000 sentences]: 94.6001118299
Accuracy in training [13928 sentences]: 92.1742134515
shoaib@MintLab ~/Dropbox/Course/NLP/HW3/pos_tagger $
```

For Q3, the accuracy is 95.41% and 93.20% on training data and test data respectively, with lowering the cases and smoothing only.
I have been able to increase the accuracy to 93.58% for the testing dataset so far, using some hard coded morphology for the unknown word.
It takes around 0.50-0.60 seconds to create the model for this question on my machine, and 8/9 minutes to run the whole prediction process. I have modified the code to show the time.



```
shoaib@MintLab ~/Dropbox/Course/NLP/HW3/pos_tagger $ python postagger.py data_pu
blic/hw3_train data_public/hw3_heldout
Elapsed time for Creating Model was 0.579418 seconds
Elapsed time for Calculating Accuracy was 146.704 seconds
Accuracy in training [13928 sentences]: 93.5824550132
Elapsed time for Calculating Accuracy was 380.916 seconds
Accuracy in training [35000 sentences]: 95.4105168275
```

How you deal with unknown words:
I have made a list of the unknown words. I have fixed some rules for CD, JJ, WDT and a few other tags. For example:
- If the word is a number I have set the tag to CD.
- If the word is a combination of numbers and alphabet, I have set the tag to JJ

Problems faced during the implementation:
The multiplication process is a bit slower, so I used log and addition, which I believe makes the program a bit faster.

For prediction using the Viterbi matrix, the runtime is slow. I was using two max() functions, and it was taking around 13/14 minutes for the whole process. I modified the code to remove one max() function, now the code runs in 8/9 minutes.

A small error analysis (what errors does your tagger make? is it hard to distinguish VBN vs. VBD?
I have checked just more than 11,000 errors. Please note that, in my following calculations, the counts contain a certain part of speech wrongly tagged as another one, or another part of speech wrongly tagged as the certain one.

In that the predictor was unable to predict between **VBN** and **VBD** 682 times. So This doesn't look that significant. However, for all the verbs:
It was unable to predict between **any verb** and **any noun** 1402 times.
It was unable to predict between **any verb** and **adjective** 665 times.
It was unable to predict between **any verb** and **other verbs, or any other parts of speech** 3193 times. This count includes the aforementioned two counts.
So I assume, verbs are confused with another verb or another part of speech somewhat evenly.

Whereas, for nouns:
It was unable to predict between **any verb** and **any noun** 1402 times.
It was unable to predict between **any noun** and **adjective** 1937 times.
It was unable to predict between **any noun** and **other nouns, or any other parts of speech** 7151 times. This count includes the aforementioned two counts.
So, a noun is more likely to be confused as other nouns, as well as other parts of speech.