ASSIGNMENT 3

• Do you see any tokenization issues?

The LAS & LDWS does not operate until the system detects white (yellow) lane lines on either the left or right.

The above sentence has tokenization issues. It is considering
LAS& ,**white(** and **yellow)** as a single word instead of splitting into different words.
Also, displaCy is not considering the full stop as a token and including with the word.

This depends on the POS tokenization rules.

• Do you think that left is usually a verb, an adjective or an adverb in the training corpus?
• Do you think the training corpus includes the tokens LAS and LDWS?

Move left to the wall on the right side of the sink where the towel holder is.

Left can be a verb, an adverb or an adjective in the training corpus.
But here in the above sentence, it is an adverb because it is modifying the verb.
Move is a verb.
Left is an adverb.

• Is it reasonable to assume that words unseen during training are proper nouns?

The training words may or may not have included LAS and LDWS tokens.
But open words that get updated constantly or frequently are categorized into unseen words.
Most of the unseen words are considered as proper nouns.

• Does the system include a language detector?

No. The system does not include a language detector. When, the french sentences are checked in displaCy, it didn’t predict the exact tag of the word.

Voici votre carte d’embarquement

For the above statement, it predicted every word as a Noun,which is practically not possible for a statement format.So it can be concluded that the system does not include a language detector.

• Do you think the letter case affects the system?

Yes.

For example: consider,

Place it on the middle shelf.

POS tags for the above statement are

Place - VERB

It - PRON

on - ADP

the - DET

middle - ADJ

shelf - NOUN

After making a few changes in the letter Cases, the predicted tags are different.

Place - VERB

It - PRON

on - ADP

the - DET

middle - ADJ

SHELF - PROPN

Not only the predicted tags are different but the grouping of words is also changed with the letter cases.

Q2: Report:

Majority Baseline Model

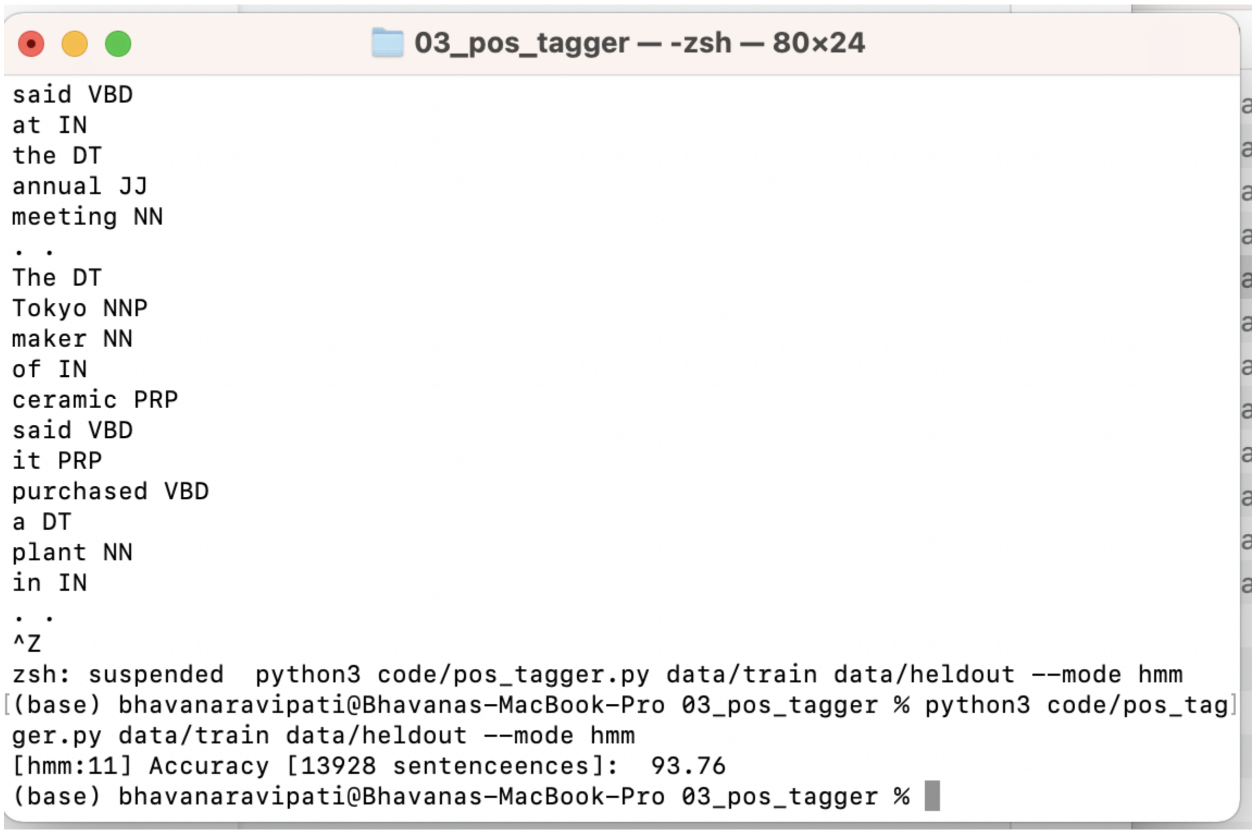
Counts of each tag is calculated and assigned the POS tag to the word which has the highest probability.Morphological rules are used to predict the tag for Unseen words like If a word ends with an s, it is tagged as NNS. .I got an accuracy of 93.83% for the test data.

HMM Model

1.Calculated likelihood and prior probability matrices

2.Previous word's tag values are stored and considered the maximum probability value of tag values to construct a viterbi matrix.

3.When unseen words are predicted using morphological context rules, accuracy (93.76%) has a rise of 1%.



03_pos_tagger — -zsh — 80×24

```
said VBD
at IN
the DT
annual JJ
meeting NN
. .
The DT
Tokyo NNP
maker NN
of IN
ceramic PRP
said VBD
it PRP
purchased VBD
a DT
plant NN
in IN
. .
^Z
zsh: suspended  python3 code/pos_tagger.py data/train data/heldout --mode hmm
[(base) bhavanaravipati@Bhavanas-MacBook-Pro 03_pos_tagger % python3 code/pos_tag]
ger.py data/train data/heldout --mode hmm
[hmm:11] Accuracy [13928 sentenceences]:  93.76
(base) bhavanaravipati@Bhavanas-MacBook-Pro 03_pos_tagger %
```

Extra Credit:

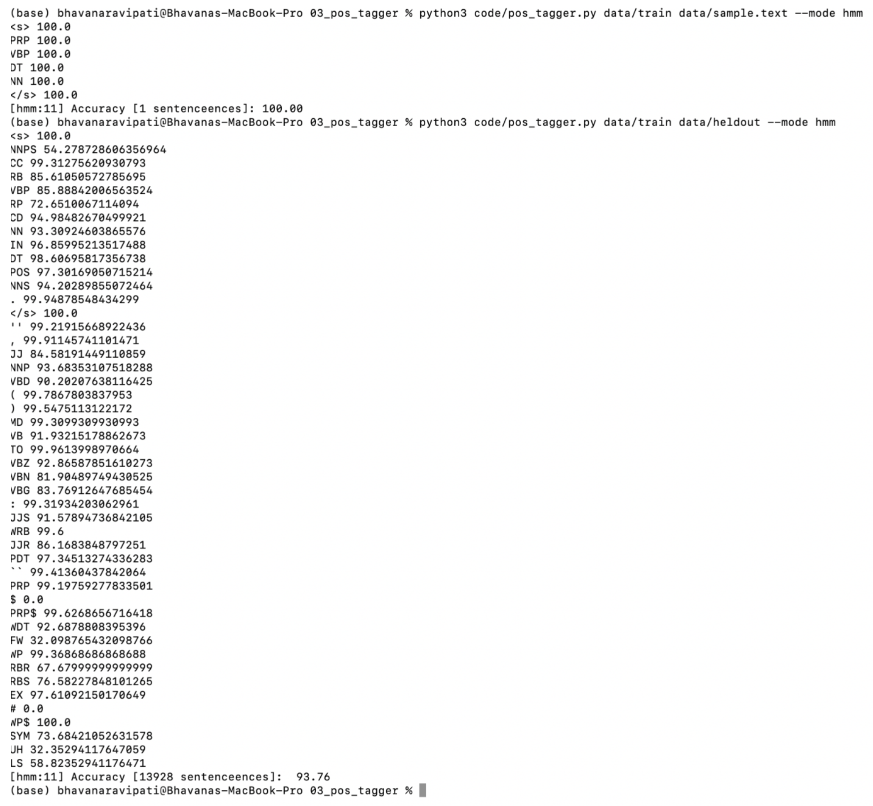
- **Calculating (and discussing) accuracy for seen and unseen words. Which ones are harder? How good are you predicting tags for unseen words?**

Calculating accuracy for unseen words is hard.Unseen words are categorized into seen words by tagging them as UNK. There is a 1.3% increase in the accuracy in the implementation.

The POS tag of an unknown word is predicted using the POS context, the word context and the substrings.Unseen words are categorized into seen words by morphological tagging.Words ending with ed are given as verbs.

- **Calculating (and discussing) accuracy per tag (or clustered tags, e.g., NOUN, VERB, ADJECTIVES, OTHER. Which tags are easy and difficult to predict? Are prepositions easy to tag?**

By calculating accuracy per tag, it is easy to predict PRP as it has the accuracy 99% and clustered tags like Noun has 93.76%.



```
(base) bhavanaravipati@Bhavanas-MacBook-Pro 03_pos_tagger % python3 code/pos_tagger.py data/train data/sample.text --mode hmm
<s> 100.0
PRP 100.0
VBP 100.0
DT 100.0
NN 100.0
</s> 100.0
[hmm:11] Accuracy [1 sentenceences]: 100.00
(base) bhavanaravipati@Bhavanas-MacBook-Pro 03_pos_tagger % python3 code/pos_tagger.py data/train data/heldout --mode hmm
<s> 100.0
NNPS 54.278728606356964
CC 99.31275620930793
RB 85.61050572785695
VBP 85.88842006563524
RP 72.6510067114094
CD 94.98482670499921
NN 93.30924603865576
IN 96.85995213517488
DT 98.60695817356738
POS 97.30169050715214
NNS 94.20289855072464
. 99.94878548434299
</s> 100.0
'' 99.21915668922436
, 99.91145741101471
JJ 84.58191449110859
NNP 93.68353107518288
VBD 90.20207638116425
( 99.7867803837953
) 99.5475113122172
MD 99.3099309930993
VB 91.93215178862673
TO 99.9613998970664
VBZ 92.86587851610273
VBN 81.90489749430525
VBG 83.76912647685454
: 99.31934203062961
JJS 91.57894736842105
WRB 99.6
JJR 86.1683848797251
PDT 97.34513274336283
`` 99.41360437842064
PRP 99.19759277833501
$ 0.0
PRP$ 99.6268656716418
WDT 92.6878808395396
FW 32.098765432098766
WP 99.36868686868688
RBR 67.67999999999999
RBS 76.58227848101265
EX 97.61092150170649
# 0.0
WP$ 100.0
SYM 73.68421052631578
UH 32.35294117647059
LS 58.82352941176471
[hmm:11] Accuracy [13928 sentenceences]:  93.76
(base) bhavanaravipati@Bhavanas-MacBook-Pro 03_pos_tagger %
```