Juhwan Lee

CS410: Natural Language Processing

Assignment 3: Parts-of-Speech Tagging

```
import nltk
from nltk.corpus import brown
from tabulate import tabulate
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('brown')
nltk.download('universal_tagset')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package brown to /root/nltk_data...
[nltk_data]   Unzipping corpora/brown.zip.
[nltk_data] Downloading package universal_tagset to /root/nltk_data...
[nltk_data]   Unzipping taggers/universal_tagset.zip.
True
```

1. (1pt) Search the web for 2 “spoof newspaper headlines”, to find such gems as: British Left Waffles on Falkland Islands, and Juvenile Court to Try Shooting Defendant. Manually tag these headlines to see if knowledge of the part-of-speech tags removes the ambiguity.

```
headline = 'British/NOUN Left/VERB Waffles/NOUN on/ADV Falkland/NOUN Islands/NOUN'
[nltk.tag.str2tuple(t) for t in headline.split()]
```

```
[('British', 'NOUN'),
 ('Left', 'VERB'),
 ('Waffles', 'NOUN'),
 ('on', 'ADV'),
 ('Falkland', 'NOUN'),
 ('Islands', 'NOUN')]
```

```
headline = 'Juvenile/NOUN Court/NOUN to/PRT Try/VERB Shooting/ADJ Defendant/NOUN'
[nltk.tag.str2tuple(t) for t in headline.split()]
```

```
[('Juvenile', 'NOUN'),
 ('Court', 'NOUN'),
 ('to', 'PRT'),
 ('Try', 'VERB'),
 ('Shooting', 'ADJ'),
 ('Defendant', 'NOUN')]
```

2. (1pt) Tokenize and tag the following sentence: They wind back the clock, while we chase after the wind. What is the output?

```
sentence = 'They wind back the clock, while we chase after the wind.'
tokens = nltk.word_tokenize(sentence)
tagged = nltk.pos_tag(tokens)
tagged
```

```
[('They', 'PRP'),
 ('wind', 'VBP'),
 ('back', 'RB'),
 ('the', 'DT'),
 ('clock', 'NN'),
 (',', ','),
 ('while', 'IN'),
 ('we', 'PRP'),
 ('chase', 'VBP'),
 ('after', 'IN'),
 ('the', 'DT'),
 ('wind', 'NN'),
 ('.', '.')]
```

3. (1pt) Pick 2 words that can be either a noun or a verb (e.g., contest). Predict which POS tag is likely to be the most frequent in the Brown

fall, dance

I think 'fall' is most frequent in the form of verb and 'dance' is most frequent in the form of noun.

```
tagged_words = brown.tagged_words(tagset='universal')
cfd = nltk.ConditionalFreqDist(tagged_words)
```

```
cfd['fall'].most_common()
```

```
[('NOUN', 71), ('VERB', 65)]
```

```
cfd['dance'].most_common()
```

```
[('NOUN', 65), ('VERB', 17)]
```

'fall' and 'dance' both were most frequent in the form of noun

4. (2pt) Use sorted() and set() to get a sorted list of tags used in the Brown corpus, removing duplicates.

```
tagged_words = brown.tagged_words()
sorted_tagged_words = sorted(tagged_words)
unique_tagged_words = set(sorted_tagged_words)
vals = [val for key, val in unique_tagged_words]
sorted(set(vals))
```

```
'VBD',
'VBD-HL',
'VBD-NC',
'VBD-TL',
'VBG',
'VBG+TO',
'VBG-HL',
'VBG-NC',
'VBG-TL',
'VBN',
'VBN+TO',
'VBN-HL',
'VBN-NC',
'VBN-TL',
'VBN-TL-HL',
'VBN-TL-NC',
'VBZ',
'VBZ-HL',
'VBZ-NC',
'VBZ-TL',
'WDT',
'WDT+BER',
'WDT+BER+PP',
'WDT+BEZ',
'WDT+BEZ-HL',
'WDT+BEZ-NC',
'WDT+BEZ-TL',
'WDT+DO+PPS',
'WDT+DOD',
'WDT+HVZ',
'WDT-HL',
'WDT-NC',
'WP$',
'WPO',
'WPO-NC',
'WPO-TL',
'WPS',
'WPS+BEZ',
'WPS+BEZ-NC',
'WPS+BEZ-TL',
'WPS+HVD',
'WPS+HVZ',
'WPS+MD',
'WPS-HL',
'WPS-NC',
'WPS-TL',
'WQL',
'WQL-TL',
```

```
 'WRB',
 'WRB+BER',
 'WRB+BEZ',
 'WRB+BEZ-TL',
 'WRB+DO',
 'WRB+DOD',
 'WRB+DOD*',
 'WRB+DOZ',
 'WRB+IN',
 'WRB+MD',
 'WRB-HL',
 'WRB-NC',
```

5. (4pt) Write programs to process the Brown Corpus and find answers to the following questions: (i) Which nouns are more common in their plural form, rather than their singular form? (Only consider regular plurals, formed with the -s suffix.) (ii) List the top 20 tags in order of decreasing frequency - what do these most frequent tags represent?

```
tagged_words = brown.tagged_words()
cfd = nltk.ConditionalFreqDist(tagged_words)
result = []
for word in set(brown.words()):
    if cfd[word+'s']['NNS'] > cfd[word]['NN']:
        result.append((word, cfd[word+'s']['NNS'], cfd[word]['NN']))

result[0:19]
```

```
[('bite', 6, 3),
 ('element', 101, 52),
 ('Pagan', 1, 0),
 ('ton', 28, 13),
 ('spike', 2, 1),
 ('affair', 62, 33),
 ('poster', 4, 3),
 ('interrelationship', 2, 1),
 ('vase', 9, 4),
 ('patron', 9, 4),
 ('Employee', 1, 0),
 ('wrestle', 1, 0),
 ('steward', 2, 1),
 ('Representative', 1, 0),
 ('Banker', 1, 0),
 ('Chestnut', 1, 0),
 ('secularist', 1, 0),
 ('theologian', 9, 5),
 ('tear', 32, 2)]
```

```
tag_list = [t for (_, t) in tagged_words]
fd = nltk.FreqDist(tag_list)
fd.most_common(20)
```

```
[('NN', 152470),
 ('IN', 120557),
 ('AT', 97959),
 ('JJ', 64028),
 ('.', 60638),
 (',', 58156),
 ('NNS', 55110),
 ('CC', 37718),
 ('RB', 36464),
 ('NP', 34476),
 ('VB', 33693),
 ('VBN', 29186),
 ('VBD', 26167),
 ('CS', 22143),
 ('PPS', 18253),
 ('VBG', 17893),
 ('PP$', 16872),
 ('TO', 14918),
 ('PPSS', 13802),
 ('CD', 13510)]
```

6. (5pt) Generate some statistics for tagged data to answer the following questions: (i) What proportion of word types are always assigned the same part-of-speech tag? (ii) How many word types are ambiguous, in the sense that they appear with at least two tags? (iii) What percentage of word tokens in the Brown Corpus involve these ambiguous word types?

```
tagged_words = brown.tagged_words(tagset='universal')
cfd = nltk.ConditionalFreqDist(tagged_words)
```

```
proportion = sum(1 for word in cfd if len(cfd[word]) == 1) / len(cfd)
proportion
```

```
0.9358510087946198
```

```
ambiguous = sum(1 for word in cfd if len(cfd[word]) > 1)
ambiguous
```

```
3596
```

```
amb_proportion = ambiguous / len(cfd)
amb_proportion
```

```
0.06414899120538024
```

7. (6pt) Write code to search the Brown Corpus for particular words and phrases according to tags, to answer the following questions: (i) Produce an alphabetically sorted list of the distinct words tagged as MD. (ii) Identify words that can be plural nouns or third person singular verbs (e.g. deals, flies). (iii) What is the ratio of masculine to feminine pronouns?

```
words = brown.words()
tagged_words = brown.tagged_words()
cfd = nltk.ConditionalFreqDist(tagged_words)
conditions = cfd.conditions()

md_words = [condition for condition in conditions if cfd[condition]['MD'] != 0]
md_words.sort()
md_words
```

```
['Can',
 'Could',
 'May',
 'Might',
 'Must',
 'Ought',
 'Shall',
 'Should',
 'Will',
 'Would',
 "c'n",
 'can',
 'colde',
 'could',
 'dare',
 'kin',
 'maht',
 'mai',
 'may',
 'maye',
 'mayst',
 'might',
 'must',
 'need',
 'ought',
 'shall',
 'should',
 'shuld',
 'shulde',
 'wil',
 'will',
 'wilt',
 'wod',
 'wold',
 'wolde',
 'would']
```

```
two_words = [condition for condition in conditions if cfd[condition]['NNS'] and cfd[condition]['VBZ']]
two_words.sort()
two_words
```

```
 'stems',
```

```
 'steps',
 'sticks',
 'stops',
 'stresses',
 'stretches',
 'strikes',
 'struggles',
 'studies',
 'subjects',
 'suits',
 'sums',
 'supplies',
 'supports',
 'surveys',
 'switches',
 'swoops',
 'talks',
 'tastes',
 'terms',
 'tests',
 'thrusts',
 'ties',
 'times',
 'tires',
 'tops',
 'tortures',
 'totals',
 'touches',
 'towers',
 'toys',
 'traces',
 'trades',
 'trains',
 'transfers',
 'transports',
 'traps',
 'travels',
 'treats',
 'tries',
 'trusts',
 'turns',
 'upsets',
 'urges',
 'uses',
 'values',
 'views',
 'visits',
 'votes',
 'vows',
 'walks',
 'wants',
 'watches',
 'weights',
 'winds',
 'wins',
 'wishes',
 'wonders',
 'works',
```

```
fd = nltk.FreqDist(words)
masc_fem_proportion = (fd['he'] + fd['He']) / (fd['she'] + fd['She'])
masc_fem_proportion
```

```
3.3384615384615386
```

8. (6pt) How serious is the sparse data problem? Investigate the performance of n-gram taggers as n increases from 1 to 6. Tabulate the accuracy score.

```
tagged_sents = brown.tagged_sents()
size = int(len(tagged_sents) * 0.9)
train_sents = tagged_sents[:size]
test_sents = tagged_sents[size:]

for i in range(1,7):
    ngram_tagger = nltk.NgramTagger(i, train_sents)
    print(ngram_tagger.evaluate(test_sents))
```

```
0.8849353534083527
0.3515747783994468
0.2029714381509189
0.15251147293644307
0.1402003310911339
0.1383667567737474
```

9. (6pt) There are 264 distinct words in the Brown Corpus having exactly three possible tags. (i) Print a table with the integers 1..10 in one column, and the number of distinct words in the corpus having 1..10 distinct tags in the other column. (ii) For the word with the greatest number of distinct tags, print out sentences from the corpus containing the word, one for each possible tag.

```
tagged_words = brown.tagged_words()
cfd = nltk.ConditionalFreqDist(tagged_words)

num_tags = []
for condition in cfd.conditions():
    num_tags.append((condition, len(cfd[condition])))

tags_by_num = []
for i in range(11):
    this_num = 0
    for (word, num) in num_tags:
        if num == i:
            this_num += 1
    tags_by_num.append((i, this_num))

print(tabulate(tags_by_num))
```

```
--  -----
 0      0
 1  47328
 2   7186
 3   1146
 4    265
 5     87
 6     27
 7     12
 8      1
 9      1
10      2
--  -----
```

```
most_distinct = ""
num_of_tags = 0

for (word, num) in num_tags:
    if num > num_of_tags:
        num_of_tags = num

for (word, num) in num_tags:
    if num == num_of_tags:
        most_distinct = word

most_distinct
```

```
'that'
```

```
distinct_tags = [tag for tag in cfd['that']]
taggend_sents = brown.tagged_sents()

for sent in taggend_sents:
    for (word, tag) in sent:
        for distinct_tag in distinct_tags:
            if distinct_tag == tag and (word == 'That' or word == 'that'):
                print(sent)
                distinct_tags.remove(distinct_tag)
                print("**********")
                break

```

```
[('The', 'AT'), ('Fulton', 'NP-TL'), ('County', 'NN-TL'), ('Grand', 'JJ-TL'), ('Jury', 'NN-TL'), ('said', 'VBD'), ('Fr
**********
```

```
[('Regarding', 'IN'), ("Atlanta's", 'NP$'), ('new', 'JJ'), ('multi-million-dollar', 'JJ'), ('airport', 'NN'), (',', ',
**********
[('``', '``'), ('Actually', 'RB'), (',', ','), ('the', 'AT'), ('abuse', 'NN'), ('of', 'IN'), ('the', 'AT'), ('process'
**********
[('While', 'CS'), ('the', 'AT'), ('city', 'NN'), ('council', 'NN'), ('suggested', 'VBD'), ('that', 'CS'), ('the', 'AT'
**********
[('He', 'PPS'), ('was', 'BEDZ'), ('able', 'JJ'), ('to', 'TO'), ('smell', 'VB'), ('a', 'AT'), ('bargain', 'NN'), ('--',
**********
[('According', 'IN'), ('to', 'IN'), ('the', 'AT'), ('official', 'JJ'), ('interpretation', 'NN'), ('of', 'IN'), ('the',
**********
[('He', 'PPS'), ('has', 'HVZ'), ('his', 'PP$'), ('own', 'JJ'), ('system', 'NN'), ('of', 'IN'), ('shorthand', 'NN'), ('
**********
[('Thus', 'NIL'), (',', ','), ('as', 'NIL'), ('a', 'NIL'), ('development', 'NIL'), ('program', 'NIL'), ('is', 'NIL'),
**********
[('In', 'IN'), ('of', 'IN-NC'), ('all', 'ABN-NC'), ('the', 'AT-NC'), ('suggestions', 'NNS-NC'), ('that', 'WPS-NC'), ('
**********
[('Thus', 'RB'), ('to', 'IN-NC'), ('has', 'HVZ'), ('light', 'JJ'), ('stress', 'NN'), ('both', 'ABX'), ('in', 'IN'), ('
**********
[('But', 'CC'), ('when', 'WRB'), ('to', 'TO-NC'), ('represents', 'VBZ'), ('to', 'IN-NC'), ('consciousness', 'NN-NC'),
**********
[('Factors', 'NNS-HL'), ('that', 'WPS-HL'), ('inhibit', 'VB-HL'), ('learning', 'VBG-HL'), ('and', 'CC-HL'), ('lead', '
**********
```

✓ 2초 오후 4:22에 완료됨