

# NLP LAB 9 : Building Bigram Tagger

Swetha Jenifer S (225229142) ¶

```
In [2]: import nltk
from nltk.tokenize import sent_tokenize, word_tokenize
import nltk
nltk.download('averaged_perceptron_tagger')
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data] C:\Users\1mscdsa17\AppData\Roaming\nltk_data...
[nltk_data] Package averaged_perceptron_tagger is already up-to-
[nltk_data] date!
[nltk_data] Downloading package punkt to
[nltk_data] C:\Users\1mscdsa17\AppData\Roaming\nltk_data...
[nltk_data] Package punkt is already up-to-date!
```

Out[2]: True

```
In [3]: text = word_tokenize("And now for something completely different")
nltk.pos_tag(text)
```

```
Out[3]: [('And', 'CC'),
         ('now', 'RB'),
         ('for', 'IN'),
         ('something', 'NN'),
         ('completely', 'RB'),
         ('different', 'JJ')]
```

```
In [4]: from nltk.corpus import brown
nltk.download('brown')
```

```
[nltk_data] Downloading package brown to
[nltk_data] C:\Users\1mscdsa17\AppData\Roaming\nltk_data...
[nltk_data] Package brown is already up-to-date!
```

Out[4]: True

```
In [5]: tagsen = brown.tagged_sents()
tagsen
```

```
Out[5]: [[('The', 'AT'), ('Fulton', 'NP-TL'), ('County', 'NN-TL'), ('Grand', 'JJ-TL'),
('Jury', 'NN-TL'), ('said', 'VBD'), ('Friday', 'NR'), ('an', 'AT'), ('investiga-
tion', 'NN'), ('of', 'IN'), ('Atlanta's', 'NP$'), ('recent', 'JJ'), ('primary',
'NN'), ('election', 'NN'), ('produced', 'VBD'), ('``', ''), ('no', 'AT'), ('e
vidence', 'NN'), ('"', ''), ('that', 'CS'), ('any', 'DTI'), ('irregularitie
s', 'NNS'), ('took', 'VBD'), ('place', 'NN'), ('.', '.')], [('The', 'AT'), ('ju
ry', 'NN'), ('further', 'RBR'), ('said', 'VBD'), ('in', 'IN'), ('term-end', 'N
N'), ('presentments', 'NNS'), ('that', 'CS'), ('the', 'AT'), ('City', 'NN-TL'),
('Executive', 'JJ-TL'), ('Committee', 'NN-TL'), (',', ','), ('which', 'WDT'),
('had', 'HVD'), ('over-all', 'JJ'), ('charge', 'NN'), ('of', 'IN'), ('the', 'A
T'), ('election', 'NN'), (',', ','), ('``', ''), ('deserves', 'VBZ'), ('the',
'AT'), ('praise', 'NN'), ('and', 'CC'), ('thanks', 'NNS'), ('of', 'IN'), ('th
e', 'AT'), ('City', 'NN-TL'), ('of', 'IN-TL'), ('Atlanta', 'NP-TL'), ('"',
'''), ('for', 'IN'), ('the', 'AT'), ('manner', 'NN'), ('in', 'IN'), ('which',
'WDT'), ('the', 'AT'), ('election', 'NN'), ('was', 'BEDZ'), ('conducted', 'VB
N'), ('.', '.')], ...]
```

```
In [6]: len(tagsen)
```

```
Out[6]: 57340
```

```
In [7]: br_train = tagsen[0:50000]
br_test = tagsen[50000:]
br_test[0]
```

```
Out[7]: [('I', 'PPSS'),
('was', 'BEDZ'),
('loaded', 'VBN'),
('with', 'IN'),
('suds', 'NNS'),
('when', 'WRB'),
('I', 'PPSS'),
('ran', 'VBD'),
('away', 'RB'),
(',', ','),
('and', 'CC'),
('I', 'PPSS'),
('haven't', 'HV*'),
('had', 'HVN'),
('a', 'AT'),
('chance', 'NN'),
('to', 'TO'),
('wash', 'VB'),
('it', 'PPO'),
('off', 'RP'),
('.', '.')]
```

```
In [8]: t0 = nltk.DefaultTagger('NN')  
t1 = nltk.UnigramTagger(br_train, backoff=t0)  
t2 = nltk.BigramTagger(br_train, backoff=t1)
```

```
In [9]: t2.evaluate(br_test)
```

```
Out[9]: 0.9111006662708622
```

```
In [10]: total_train = [len(l) for l in br_train]  
sum(total_train)
```

```
Out[10]: 1039920
```

```
In [11]: total_test = [len(l) for l in br_test]  
sum(total_test)
```

```
Out[11]: 121272
```

```
In [12]: t1.evaluate(br_test)
```

```
Out[12]: 0.8897849462365591
```

```
In [13]: t2.evaluate(br_test)
```

```
Out[13]: 0.9111006662708622
```

```
In [14]: br_train[0]
```

```
Out[14]: [('The', 'AT'),
          ('Fulton', 'NP-TL'),
          ('County', 'NN-TL'),
          ('Grand', 'JJ-TL'),
          ('Jury', 'NN-TL'),
          ('said', 'VBD'),
          ('Friday', 'NR'),
          ('an', 'AT'),
          ('investigation', 'NN'),
          ('of', 'IN'),
          ("Atlanta's", 'NP$'),
          ('recent', 'JJ'),
          ('primary', 'NN'),
          ('election', 'NN'),
          ('produced', 'VBD'),
          ('', ''),
          ('no', 'AT'),
          ('evidence', 'NN'),
          ('', ''),
          ('that', 'CS'),
          ('any', 'DTI'),
          ('irregularities', 'NNS'),
          ('took', 'VBD'),
          ('place', 'NN'),
          ('.', '.')]

```

```
In [15]: br_train[1277]
```

```
Out[15]: [('', ''),
          ('I', 'PPSS'),
          ('told', 'VBD'),
          ('him', 'PPO'),
          ('who', 'WPS'),
          ('I', 'PPSS'),
          ('was', 'BEDZ'),
          ('and', 'CC'),
          ('he', 'PPS'),
          ('was', 'BEDZ'),
          ('quite', 'QL'),
          ('cold', 'JJ'),
          ('.', '.')]

```

```
In [16]: br_train[1277][11]
```

```
Out[16]: ('cold', 'JJ')
```

```
In [17]: br_train_flat = [(word, tag) for sent in br_train for (word, tag) in sent]
```

```
In [18]: br_train_flat[:40]
```

```
Out[18]: [('The', 'AT'),
('Fulton', 'NP-TL'),
('County', 'NN-TL'),
('Grand', 'JJ-TL'),
('Jury', 'NN-TL'),
('said', 'VBD'),
('Friday', 'NR'),
('an', 'AT'),
('investigation', 'NN'),
('of', 'IN'),
("Atlanta's", 'NP$'),
('recent', 'JJ'),
('primary', 'NN'),
('election', 'NN'),
('produced', 'VBD'),
('``', ''),
('no', 'AT'),
('evidence', 'NN'),
(''''', ''),
('that', 'CS'),
('any', 'DTI'),
('irregularities', 'NNS'),
('took', 'VBD'),
('place', 'NN'),
('.', '.'),
('The', 'AT'),
('jury', 'NN'),
('further', 'RBR'),
('said', 'VBD'),
('in', 'IN'),
('term-end', 'NN'),
('presentments', 'NNS'),
('that', 'CS'),
('the', 'AT'),
('City', 'NN-TL'),
('Executive', 'JJ-TL'),
('Committee', 'NN-TL'),
(',', ','),
('which', 'WDT'),
('had', 'HVD')]
```

```
In [19]: br_train_flat[13]
```

```
Out[19]: ('election', 'NN')
```

```
In [20]: fd = nltk.FreqDist(br_train_flat)
cfd = nltk.ConditionalFreqDist(br_train_flat)
```

```
In [21]: cfd['cold'].most_common()
```

```
Out[21]: [('JJ', 110), ('NN', 8), ('RB', 2)]
```

```
In [22]: br_train_2grams = list(nltk.ngrams(br_train_flat, 2))
br_train_cold = [a[1] for (a,b) in br_train_2grams if b[0] == 'cold']
fdist = nltk.FreqDist(br_train_cold)
[tag for (tag, _) in fdist.most_common()]
```

```
Out[22]: ['AT',
'IN',
'CC',
'QL',
'BEDZ',
'JJ',
',',
'DT',
'PP$',
'RP',
'',
'NN',
'VBN',
'VBD',
'CS',
'BEZ',
'DOZ',
'RB',
'PPSS',
'BE',
'VB',
'VBZ',
'NP$',
'BEDZ*',
'--',
'DTI',
'WRB',
'BED']
```

```
In [23]: br_pre = [(w2+"/"+t2, t1) for ((w1,t1),(w2,t2)) in br_train_2grams]
br_pre_cfd = nltk.ConditionalFreqDist(br_pre)
br_pre
```

```
Out[23]: [('Fulton/NP-TL', 'AT'),
('County/NN-TL', 'NP-TL'),
('Grand/JJ-TL', 'NN-TL'),
('Jury/NN-TL', 'JJ-TL'),
('said/VBD', 'NN-TL'),
('Friday/NR', 'VBD'),
('an/AT', 'NR'),
('investigation/NN', 'AT'),
('of/IN', 'NN'),
('Atlanta's/NP$', 'IN'),
('recent/JJ', 'NP$'),
('primary/NN', 'JJ'),
('election/NN', 'NN'),
('produced/VBD', 'NN'),
('`/``', 'VBD'),
('no/AT', '``'),
('evidence/NN', 'AT'),
(''''/'', 'NN'),
('that/CS', '""'),
('DTI', 'DTI'),
('DTI', 'DTI')]
```

```
In [24]: br_pre_cfd['cold/NN'].most_common()
```

```
Out[24]: [('AT', 4), ('JJ', 2), ('', 1), ('DT', 1)]
```

```
In [25]: br_pre_cfd['cold/JJ'].most_common()
```

```
(('', 2),
('``', 2),
('NN', 2),
('VBN', 2),
('VBD', 2),
('CS', 1),
('BEZ', 1),
('DOZ', 1),
('RB', 1),
('PPSS', 1),
('BE', 1),
('VB', 1),
('VBZ', 1),
('NP$', 1),
('BEDZ*', 1),
('--', 1),

('RP', 1),
('DTI', 1),
('WRB', 1),
('BED', 1)]
```

```
In [26]: bigram_tagger = nltk.BigramTagger(br_train)
```

```
In [27]: text1 = word_tokenize('I was very cold.')
bigram_tagger.tag(text1)
```

```
Out[27]: [('I', 'PPSS'), ('was', 'BEDZ'), ('very', 'QL'), ('cold', 'JJ'), ('.', '.')]


```

```
In [28]: text2 = word_tokenize('I had a cold.')
bigram_tagger.tag(text2)
```

```
Out[28]: [('I', 'PPSS'), ('had', 'HVD'), ('a', 'AT'), ('cold', 'JJ'), ('.', '.')]


```

```
In [29]: text3 = word_tokenize('I had a severe cold.')
bigram_tagger.tag(text3)
```

```
Out[29]: [('I', 'PPSS'),
          ('had', 'HVD'),
          ('a', 'AT'),
          ('severe', 'JJ'),
          ('cold', 'JJ'),
          ('.', '.')]


```

```
In [30]: text4 = word_tokenize('January was a cold month.')
bigram_tagger.tag(text4)
```

```
Out[30]: [('January', None),
          ('was', None),
          ('a', None),
          ('cold', None),
          ('month', None),
          ('.', None)]


```

```
In [31]: text5 = word_tokenize('I failed to do so.')
bigram_tagger.tag(text5)
```

```
Out[31]: [('I', 'PPSS'),
          ('failed', 'VBD'),
          ('to', 'TO'),
          ('do', 'DO'),
          ('so', 'RB'),
          ('.', '.')]


```

```
In [32]: text6 = word_tokenize('I was happy, but so was my enemy.')
bigram_tagger.tag(text6)
```

```
Out[32]: [('I', 'PPSS'),
          ('was', 'BEDZ'),
          ('happy', 'JJ'),
          (',', ','),
          ('but', 'CC'),
          ('so', 'RB'),
          ('was', 'BEDZ'),
          ('my', 'PP$'),
          ('enemy', 'NN'),
          ('.', '.')]


```



```
In [33]: text7 = word_tokenize('So, how was the exam?')
bigram_tagger.tag(text7)
```

```
Out[33]: [('So', 'RB'),
          (',', ', ', ', '),
          ('how', 'WRB'),
          ('was', 'BEDZ'),
          ('the', 'AT'),
          ('exam', None),
          ('?', None)]
```

```
In [34]: text8 = word_tokenize('The students came in early so they can get good seats.')
bigram_tagger.tag(text8)
```

```
Out[34]: [('The', 'AT'),
          ('students', 'NNS'),
          ('came', 'VBD'),
          ('in', 'IN'),
          ('early', 'JJ'),
          ('so', 'CS'),
          ('they', 'PPSS'),
          ('can', 'MD'),
          ('get', 'VB'),
          ('good', 'JJ'),
          ('seats', 'NNS'),
          ('.', '.')] 
```

```
In [35]: text10 = word_tokenize('That was so incredible.')
bigram_tagger.tag(text10)
```

```
Out[35]: [('That', 'DT'),
          ('was', 'BEDZ'),
          ('so', 'QL'),
          ('incredible', 'JJ'),
          ('.', '.')] 
```

```
In [ ]:
```