Aashish Yadavally
Fall 2019

Homework 4

CSCI 6900
September 12, 2019

I wrote a :class: ‘POSTagger‘ wrapped around NLTK utilities in *hw4.py* to minimize code-duplication in solving Questions 1-4 in the assignment, and also to perform the corresponding analysis. The code snippets below use this :class: ‘POSTagger‘ to show the results, as recorded.

Question 1

The following code snippet helps build a bigram tagger with two back-off models where the default tagger assigns ’NN’ by default.

```
>>> from hw4 import POSTagger
>>> pt = POSTagger(corpus='brown')
>>> pt.evaluate()
Accuracy is 0.8452108043456593
Performing analysis...
Frequency Distribution of wrong predictions...
FreqDist({('to', 'TO', 'IN'): 80, ('libraries', 'NN', 'NNS'): 22,
('Belgians', 'NN', 'NPS'): 20, ('as', 'CS', 'QL'): 14,
('that', 'CS', 'DT'): 11, ('Lumumba', 'NN', 'NP'): 10,
('turnpikes', 'NN', 'NNS'): 10, ('Holmes', 'NN', 'NP'): 9,
('it', 'PPS', 'PPO'): 9, ('that', 'CS', 'WPS'): 8, ...})
```

It can be observed that on the test set consisting of 463 instances, an accuracy of 84.52 % was achieved. The most mispredicted word was ’to’ which is actually listed as a preoposition, but was mispredicted as an infinitive 80 times. As an infinitive, ’to’ is used before infinitive verbs, while as a preposition, it is used to signify ’for’ or ’towards’ and is often followed by a noun construction. In theory, there should be an improvement in the recognition of parts of speech of the word ’to’ when using trigrams, because the knowledge of parts of speech of the preceding and succeeding words simultaneously should help avoid the confusion that the tagger was facing in making decisions using a bigram tagger. Even the second word from the wrongly tagged words, i.e, ’libraries’ was wrongly tagged as ’NN’ (common noun) while the actual tag was ’NNS’, i.e, pluran noun. Thus in this case, the tense of the word wasn’t taken into account by the tagger. Similarly, ’Belgians’ was tagged more commonly as noun while it was listed as a plural proper noun, which is more descriptive. By looking at these mistagged words, it can be concluded that the tagger is failing to predict descriptive tags by taking into account the ’singular’ or ’plural’ form, as was originally listed in the corpus.

From the following code snippet, the cases where ’that’ was mistagged can be evaluated:

```
>>> from hw4 import POSTagger
>>> pt = POSTagger(corpus='brown')
>>> fd = pt.evaluate()
>>> for i in fd:
...     if i[0] == 'that':
...     print(i, fd[i])
('that', 'CS', 'DT') 11
('that', 'CS', 'WPS') 8
```

It can be observed that 'that' was mistagged 8 times as a WH-pronoun and 11 times as a determiner/pronoun while it was originally listed as a subordinating conjunction. 'That' could be a determiner if it is followed by a noun and cannot be a WH-pronoun. In particular, it can be determined successfully as a conjunction if the tagger identifies that it's joining two clauses, for which, it will need more context information of the word.

Question 2(a)

The following code snippet helps build a bigram tagger on the BNC News Corpus with two back-off models where the default tagger assigns 'NN' by default. In this case, CLAWS5 tagset is used for the British National Corpus.

```
>>> from hw4 import POSTagger
>>> pt = POSTagger(corpus='bnc')
>>> pt.evaluate()
Accuracy is 0.9347007124677158
Performing analysis...
Frequency Distribution of wrong predictions...
FreqDist({('to', 'PRT', 'ADP'): 7002, ('in', 'ADP', 'X'): 841,
('as', 'CONJ', 'ADP'): 650, ('her', 'DET', 'PRON'): 614,
('on', 'ADP', 'X'): 568, ('more', 'ADV', 'DET'): 560,
('that', 'CONJ', 'X'): 556, ('that', 'CONJ', 'DET'): 488,
('to', 'ADP', 'PRT'): 456, ('after', 'ADP', 'CONJ'): 381, ...})
```

The performance of the BNC-CLAWS5 is much better than BROWN on a tagger built similarly, mainly because the BROWN tagset is much more descriptive, thus with more class distinctions. The CLAWS5 tagset has around 60 classes, which was being used to distinguish around 500,000 instances whereas BROWN tagset has around 230 classes being used to classify around 4,500 instances. Thus, BNC-CLAWS5 tagset ends up performing better.

Question 2(b)

The following code snippet helps build a bigram tagger on the BNC News Corpus with two back-off models where the default tagger assigns 'NN' by default. In this case, 'universal' tagset is used for the British National Corpus.

```
>>> from hw4 import POSTagger
>>> pt = POSTagger(corpus='bnc', tagset='universal')
>>> pt.evaluate()
Accuracy is 0.9284675323050359
Performing analysis...
Frequency Distribution of wrong predictions...
FreqDist({('to', 'TO0', 'PRP'): 6640, ('in', 'PRP', 'PRP-AVP'): 602,
('her', 'DPS', 'PNP'): 585, ('to', 'PRP', 'TO0'): 540,
('on', 'PRP', 'AVP'): 518, ('as', 'CJS', 'PRP'): 515,
('more', 'AV0', 'DT0'): 429, ('on', 'PRP', 'PRP-AVP'): 407,
('that', 'CJT', 'DT0'): 395, ('in', 'PRP', 'AVP'): 385, ...})
```

The performance of the BNC-UNIVERSAL is only marginally better than BNC-CLAWS5 on a tagger built similarly. The CLAWS5 tagset has around 60 classes, which was being used to distinguish around 500,000 instances whereas UNIVERSAL tagset has around 20 classes being used to classify around the same number of instances. In my opinion, in such a case, there is a tradeoff between descriptiveness of the tagger and the performance, which in this case is not very different.

Question 3

In Section 5 of NLTK book, it is shown that an increase in the size of the model results in the rapid improvement in the performance of the tagger. This can also be observed the difference in performance between the same kind of tagger in Table 1 and Table 2, wherein the size of the Brown corpus is around 4500 tagged words, and that of BNC corpus is around 500,000 tagged words. Also, it is shown that by increasing the context in the data, i.e, by using the n-gram taggers, there should be an improvement in the performance. This can easily be seen in all the three kinds of taggers (observe increase in performance vertically down in each of the tables) on both the Brown and BNC corpora, even while using different tagsets.

Table 1 records the performance of default tagger 'NN', UnigramTagger, BigramTagger, TrigramTagger, trained on the first 90% of the Brown Corpus, and tested on last 10% of the corpus.

Table 1: Performance on Brown Corpus

Tagger	Accuracy
Default	12.6 %
Unigram	83.6 %
Bigram	84.52 %
Trigram	84.53 %

Table 2: Performance on BNC Corpus with CLAWS5 Tagset

Tagger	Accuracy
Default	0.0 %
Unigram	89.15 %
Bigram	92.85 %
Trigram	92.953 %

Similarly, Table 2 and Table 3 record the performance of each of these taggers with BNC News Corpus using CLAWS5 tagset and Universal tagset respectively.

Tagger	Accuracy
Default	0.0 %
Unigram	91.57 %
Bigram	93.47 %
Trigram	93.474 %

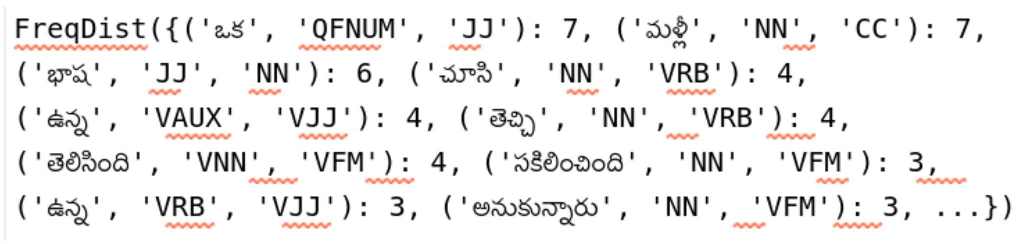
Table 3: Performance on BNC Corpus with Universal Tagset

In each of the tables, the default tagger assigning 'NN' performs the worst. In the case of the BNC corpus, NN should have been replaced by 'NN0' in the BNC_CLAWS5 tagset, which is the tag for a noun, and by 'NOUN' in the UNIVERSAL tagset.

Question 4

One of the tagged corpora in NLTK in a language other than English is 'indian', which contains tagged sentences from Bangla, Marathi, Telugu, Hindi. For this question, I chose to build a bigram parts-of-speech tagger for Telugu (my mother tongue) language, and analyze it's performance.

```
>>> from hw4 import POSTagger
>>> pt = POSTagger(corpus='indian', lang='telugu')
>>> pt.evaluate()
Accuracy is 0.6617647058823529
Performing analysis...
Frequency Distribution of wrong predictions...
```



```
FreqDist({('ఒక', 'QFNUM', 'JJ'): 7, ('మళ్లీ', 'NN', 'CC'): 7,
('భాష', 'JJ', 'NN'): 6, ('చూసి', 'NN', 'VRB'): 4,
('ఉన్న', 'VAUX', 'VJJ'): 4, ('తెచ్చి', 'NN', 'VRB'): 4,
('తెలిసింది', 'VNN', 'VFM'): 4, ('సకిలించింది', 'NN', 'VFM'): 3,
('ఉన్న', 'VRB', 'VJJ'): 3, ('అనుకున్నారు', 'NN', 'VFM'): 3, ...})
```

In comparison, the accuracies for the other languages are:

- Bangla : 0.6688218390804598
- Marathi : 0.7393284006829823
- Hindi : 0.8186753528773073