Lab 2 TDT4310

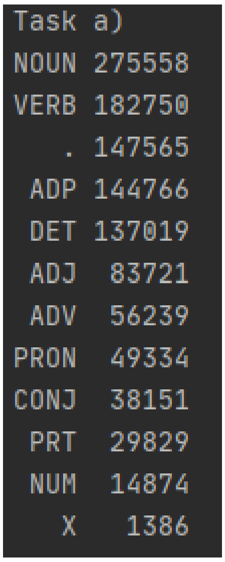
Vebjørn Ohr

February 18 2021

Exercise 1

Task a)

The NLTK FreqDist method was used to find the most common tags used in the Brown corpus. The universal tagset was used for more readability.



```
Task a)
NOUN 275558
VERB 182750
   . 147565
 ADP 144766
 DET 137019
 ADJ  83721
 ADV  56239
PRON  49334
CONJ  38151
 PRT  29829
 NUM  14874
   X   1386
```

Figur 1: Task 1 a)

Task b)

The NLTK ConditionalFreqDist method was used to find the distribution of tags for each word. Those with two or more tags were added to a set to find all ambiguous words. This resulted in 3408 ambiguous words.

Task c)

The fraction of ambiguous words was simply calculated by using the same set of words from task b), and dividing it by the total number of words in the Brown corpus. This resulted in 6.08 percent ambiguous words.

Task d)

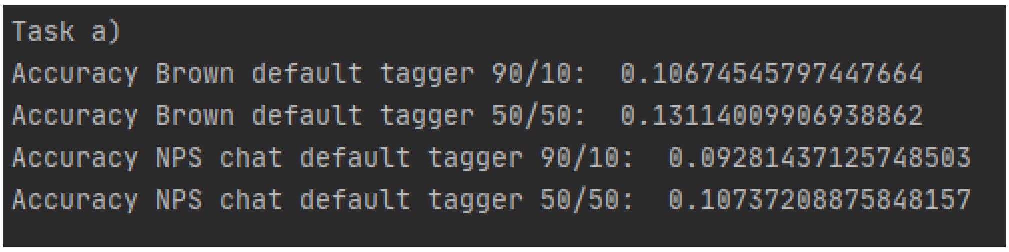
The ConditionalFreqDist from task b) was used to count the number of tags for each word. They were done sorted by the number of tags. The sentences of the Brown corpus was then iterated through, looking for sentences containing the word. I chose to print the first four sentences found for each of the 10 words. For example, the word down"had 6 different tags, and from the first four sentences it was used for as particle (PRT) and adverb (ADV).

Exercise 2

First the data was divided into training and test data for both the Brown and the NPS Chat corpora, in both 90/10 and 50/50 fractions.

Task a)

To find the most common tags for each corpus the same method as in task 1 a) was used, by utilizing the NLTK FreqDist function. For the brown corpus this was found to be nouns (NN) and for the NPS Chat corpora it was interjections (UH). The accuracies are shown in Figure 2. The default tagger doesn't use any training, so the accuracies are simply the fraction of the words that fit the default tag, and it seems like with a smaller test set there is a smaller fraction of these tags.

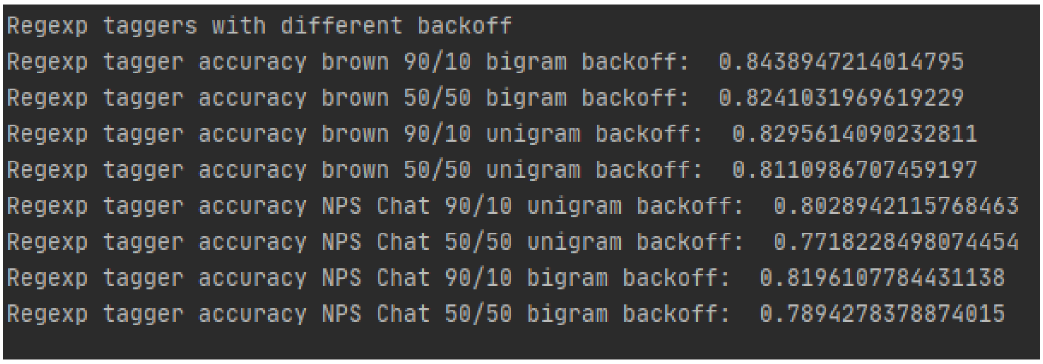


```
Task a)
Accuracy Brown default tagger 90/10:  0.10674545797447664
Accuracy Brown default tagger 50/50:  0.13114009906938862
Accuracy NPS chat default tagger 90/10:  0.09281437125748503
Accuracy NPS chat default tagger 50/50:  0.10737208875848157
```

Figur 2: Task 2 a)

Task b)

The regular expression pattern was taken from the NLTK book chapter 5, and removing the default tag and instead using backoff taggers. I chose to implement the bigram taggers with a unigram backoff, which again had default tagger as backoff. The bigram tagger had the highest accuracy when used alone, and also as backoff for the regexp tagger for both corpora. Overall the accuracy was also slightly lower using the 50/50 split, which makes sense as there is less data to train on. The regexp tagger's accuracy was lower than bigram/unigram which is probably because there are many exceptions to the regular expression patterns. The bigram tagger's accuracy is also caused by the unigram backoff, and no backoff would yield lower accuracy as there would be fewer words fitting the model.



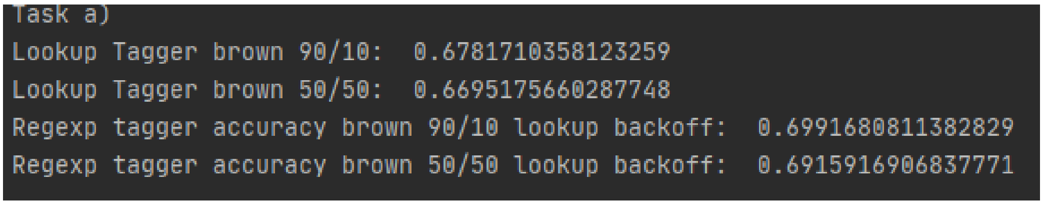
```
Regexp taggers with different backoff
Regexp tagger accuracy brown 90/10 bigram backoff:  0.8438947214014795
Regexp tagger accuracy brown 50/50 bigram backoff:  0.8241031969619229
Regexp tagger accuracy brown 90/10 unigram backoff:  0.8295614090232811
Regexp tagger accuracy brown 50/50 unigram backoff:  0.8110986707459197
Regexp tagger accuracy NPS Chat 90/10 unigram backoff:  0.8028942115768463
Regexp tagger accuracy NPS Chat 50/50 unigram backoff:  0.7718228498074454
Regexp tagger accuracy NPS Chat 90/10 bigram backoff:  0.8196107784431138
Regexp tagger accuracy NPS Chat 50/50 bigram backoff:  0.7894278378874015
```

Figur 3: Task 2 b)

Exercise 3

Task a)

The lookup tagger was made as in the NLTK book chapter 5, using the 200 most frequent words and their like likely tags to create a unigram tagger. I used a default tagger as backoff to make the results more comparable with those from Exercise 2. The accuracies are lower using the lookup tagger, which is caused by only using the 200 most common words instead of the entire corpus as in exercise 2.

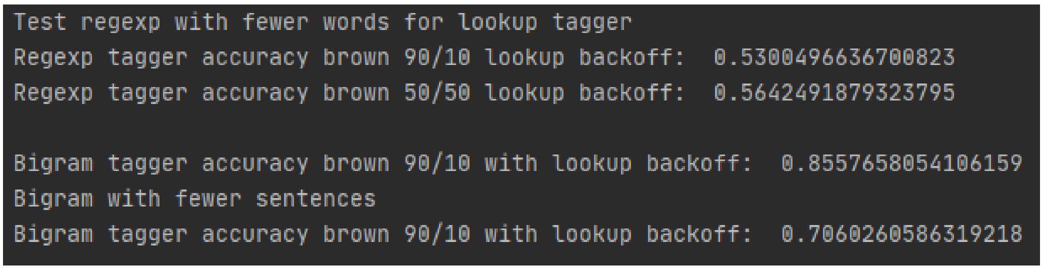


```
Task a)
Lookup Tagger brown 90/10:  0.6781710358123259
Lookup Tagger brown 50/50:  0.6695175660287748
Regexp tagger accuracy brown 90/10 lookup backoff:  0.6991680811382829
Regexp tagger accuracy brown 50/50 lookup backoff:  0.6915916906837771
```

Figur 4: Task 3 a)

Task b)

It can be seen that the accuracies fall when the size of the training data falls. This makes sense because there will be more examples the model has not encountered. This holds true both for the regexp and the bigram taggers which were tested in this case with 1000 words or sentences.



```
Test regexp with fewer words for lookup tagger
Regexp tagger accuracy brown 90/10 lookup backoff:  0.5300496636700823
Regexp tagger accuracy brown 50/50 lookup backoff:  0.5642491879323795

Bigram tagger accuracy brown 90/10 with lookup backoff:  0.8557658054106159
Bigram with fewer sentences
Bigram tagger accuracy brown 90/10 with lookup backoff:  0.7060260586319218
```

Figur 5: Task 3 b)

Exercise 4

Task a)

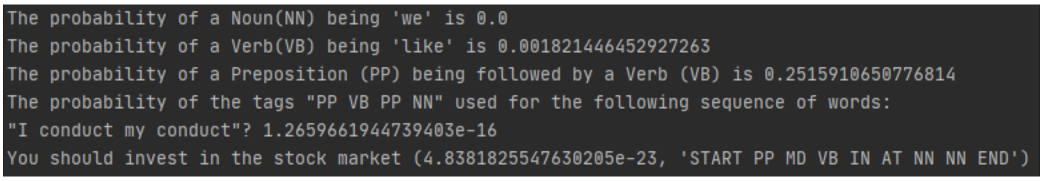
The probabilities were found using the provided probDist function, and the returned ConditionalProbDist returned. The tags distribution was used to find probabilities of a tag following another tag, and the tagwords distribution to find the probability of a tag being a specific word. The probability of ’we’ being a noun was calcualted as 0, which makes sense as we is not a noun. The probability of a verb being ’like’ was 0.18 percent, and the probability of a preposition being followed by a verb was 25 percent.

Task b)

The same probability distributions was used, multiplying the probabilities for each word and tag. This resulted in a near 0 proability, which is not surprising considering ’conduct’ is ambigouos.

Task c)

The distinct tags was found for the corpus, and then the provided Viterbi function was used to find the best tags for the sentence ”You should invest in the stock market”. This turned out be PP MD VB IN AT NN NN”.



```
The probability of a Noun(NN) being 'we' is 0.0
The probability of a Verb(VB) being 'like' is 0.001821446452927263
The probability of a Preposition (PP) being followed by a Verb (VB) is 0.2515910650776814
The probability of the tags "PP VB PP NN" used for the following sequence of words:
"I conduct my conduct"? 1.2659661944739403e-16
You should invest in the stock market (4.8381825547630205e-23, 'START PP MD VB IN AT NN NN END')
```

Figur 6: Task 4