Reilly McBride

Period 5

6 March 2017

NLTK Exercises: Chapter 7

Exercise 3: *Pick one of the three chunk types in the CoNLL corpus. Inspect the CoNLL corpus and try to observe any patterns in the POS tag sequences that make up this kind of chunk. Develop a simple chunker using the regular expression chunker nltk.RegexpParser. Discuss any tag sequences that are difficult to chunk reliably.*

Chunk type chosen: NP

Typical POS Sequences, determined by observing outputs of:

print(nltk.corpus.conll2000.chunked\_sents('train.txt', chunk\_types=['NP'])[INDEX])

Through observation, common POS sequences are: DT NNP; NNP NNP; DT NN JJ NN. Others seen are typically variations of these, such as just NNP, or DT JJ NN. Before JJ or NN, you often see PP$. My initial code was:

**def** three\_seven():  
 test\_sentences = (nltk.corpus.conll2000.chunked\_sents(**'train.txt'**, chunk\_types=[**'NP'**]))  
 grammar = **r"""  
 NP: {<NNP>+}  
 {<DT|PP\$>?<NN>\*<JJ>\*<NN>}   
 """** cp = nltk.RegexpParser(grammar)  
 chunkscore = cp.evaluate(test\_sentences)  
 **for** c **in** chunkscore.incorrect():  
 print(c)

This code incorrectly chunked a lot of chunks that were either chunks with NNP or NN. On any given sentence however, I could see successfully chunked NNPs and NNs. It missed chunks that were NNS (plural nouns) and contained comparative adjectives (JJR). This was improved significantly (nearly 10% precision increase) by re-writing grammar as:

**r"""  
 NP:   
 {<DT|PP\$>?<NN|NNS>\*<JJ|JJR>\*<NN|NNS>}   
 {<NNP>+}   
 {<NN>+}  
 """**

The chunker was still missing chunks containing CD (Cardinal Number), and DT with NN or NNP. The alterations shown below a 5% precision increase (to be at 64.9%), although CD and possessive endings (POS) still caused many missed chunks.

grammar = **r"""  
 NP:   
 {<DT|PP\$>?<NN|NNS>\*<JJ|JJR>\*<CD>\*<NN|NNS>}   
 {<DT|PP\$>?<NNP>+}   
 {<DT|PP\$>?<NN>+}  
 """**

Exercise 7*: Carry out the following evaluation tasks for any of the chunkers you have developed earlier. (Note that most chunking corpora contain some internal inconsistencies, such that any reasonable rule-based approach will produce errors.)*

* *Evaluate your chunker on 100 sentences from a chunked corpus, and report the precision, recall and F-measure.*
* *Use the chunkscore.missed() and chunkscore.incorrect() methods to identify the errors made by your chunker. Discuss.*
* *Compare the performance of your chunker to the baseline chunker discussed in the evaluation section of this chapter.*

I used the following chunker, which I developed based on the discussion in exercise 3.

test\_sents = nltk.corpus.conll2000.chunked\_sents(**'test.txt'**, chunk\_types=[**'NP'**])[0:100]  
  
 grammar = **r"""  
 NP:   
 {<DT|PP\$>?<NN|NNS>\*<JJ|JJR>\*<CD>\*<NN|NNS>}   
 {<DT|PP\$>?<JJ>?<NNP>+<CD>?<JJ>?<NN>?}   
 {<DT|PP\$>?<JJ>?<NN>+}  
 {<PRP>+}  
 {<WP>+}  
 """** chunker = nltk.RegexpParser(grammar)  
  
 chunkscore = chunker.evaluate(test\_sents)  
 print(chunkscore)

*On the first 100 sentences, this was the chunkscore result:*

ChunkParse score:

IOB Accuracy: 77.2%%

Precision: 60.8%%

Recall: 62.4%%

F-Measure: 61.6%%

*On all the sentences, this was the result:*

ChunkParse score:

IOB Accuracy: 83.2%%

Precision: 67.1%%

Recall: 67.8%%

F-Measure: 67.4%%

Based on chunkscore.missed() and chunkscore.incorrect(), my chunker was missing chunks that included CD (Cardinal Numbers) and POS (possessive noun endings), as well as a lot of NNP (Proper Noun), and even some basic chunks that should’ve followed the format listed in the rules. Several attempts to fix the CD issue actually lowered the ChunkParse score. It seems that being too specific in the rules leads to inaccuracy in creating other chunks, or makes the chunker somehow too specific to capture a lot of chunks in smaller sections, compared to the whole corpus. I’m not sure why it would’ve missed chunks that aligned with the specified POS tag sequences, although perhaps those were related to the aforementioned internal inconsistencies, or odd patterns before those chunks that caused parsing errors.

Compared to the baseline chunkers given in the evaluation section, my chunker was not as effective (in every category, with similar differences in percentages in each case). Attempts I made to combine more of their given rules with my chunker made it less effective. Based on observation of the missed and incorrect chunkscores when running the code of the chunkers they used, there were a lot of similarities in what the different chunkers missed (in terms of POS tags), although obviously the baseline chunkers missed less overall.