NLP 1 Pen & paper exercises - Ard Snijders - 12854913

1

(i) A: Your dog's chasing a man on a bicycle.
B: Don't be silly, my dog can't ride a bicycle.

Syntactic structure ambiguity: without knowledge of the world, difficult to tell whether dog or man is on a bicycle

(ii) Irritated lady in a post office: Must I stick the stamp on myself? Post-office employee: I think you'll accomplish more, madam, if you stick it on the package.

Parts of speech ambiguity; without knowledge of the world, difficult to tell whether stamp is stuck on package or madam.

(iii) Customer: Will my burger be long? Waiter: No sir, it will be round and flat.

Word ambiguity: long can refer to both duration (adverb) and shape (adjective)

(iv) What did the barman say to the ghost? Sorry sir, we don't serve spirits here.

Word ambiguity: spirits can refer to both ghosts and alcoholic beverages (both nouns).

(b)

The bigram model assumes that the probability of a tag depends on the preceding tag, or:

$$\widehat{t}_1^n = argmax \prod_{i=1}^{n} P(w_i | t_i) P(t_i | t_{i-1})$$

That is, the estimated probability of a tag t can be modelled as the dot product of:

- the probability of a word given that tag t
- the probability of t given another tag t preceding it

(c) Given the following training data, show the estimates that would be obtained for the probabilities in the equation you gave: [4 points]

```
the_DT0 green_AJ0 bottle_NN1 leaked_VVD ._PUN the_DT0
suppliers_NN2 bottle_VVB water_NN1 ._PUN green_AJ0 water_NN1
suppliers_NN2 bottle_VVB ._PUN
```

T sequences	Occurrences	Probabilities	WT sequences	Occurrences	Probabilities
DTO	2	/	the_DTO	2	2/12
DTO AJO	1	1/2	green_AJO	2	2/12
DTO NN2	1	1/2	bottle_NN1	1	1/12
AJO	2	/	leaked_VVD	1	1/12
AJO NN1	2	2/2 = 1	suppliers_NN2	2	2/12
NN1	3	/	bottle_VVB	2	2/12
NN1 VVD	1	1/3	water_NN1	2	2/12
NN1 PUN	1	1/3	.pun	3	3/12
NN1 NN2	1	1/3			
VVD	1	/			
VVD PUN	1	1/1 = 1			
PUN	3	/			
PUN DTO	1	1/3			
PUN AJO	1	1/3			
PUN </S>	1	1/3			
NN2	2	/			
NN2 VVB	2	2/2 = 1			
VVB	2	/			
VVB NN1	1	1/2			
VVB PUN	1	1/2			

(d) Explain what is meant by the terms smoothing and backoff in the context of stochastic POS tagging. [1 point]

It is possible that certain words or phrases were not seen before, and these "non-existent" events would therefore evaluate to *zero-probability estimates.* In the context of *stochastic* POS tagging, this would present problems in calculating probabilities.

Smoothing and backoff are two different methods that can be used to deal with these phenomena.

Specifically, smoothing is used to ensure that unseen words or phrases have a non-zero probability. This is achieved by redistributing the probability mass from more common events to rare or unknown ones. For instance, when calculating conditional probabilities, k-smoothing or laplacian smoothing works by adding a term k to the numerator and the length of the vocabulary to the denominator, to ensure that even unseen events do not evaluate to zero.

Backoff refers to the practice of using more/less context when appropriate, approximating unseen events with more general probabilities. For instance, one can employ tri-grams if there is sufficient evidence for using them, but if this is not the case it might be more appropriate to *fall back* onto the bi-gram form and forego context/specificity in favour of generalization. In other words, if there are no counts for a particular n-gram (= no evidence), we fall back to the (N-1)-gram until we reach a history with some counts.

(e) One common source of errors in stochastic POS taggers is that nouns occurring immediately before other nouns (e.g. catamaran trailer) are often tagged as adjectives and, conversely, prenominal adjectives are often tagged as nouns (e.g.trial offer). Suggest possible reasons for this effect. [4 points]

I am assuming that the POS tagger tags words based on both lexical probability (what is the most likely POS for term x?) and tag sequence probability (what is the most likely POS tag, given that [POS TAG] occurs after it?)

Catamaran is a relatively uncommon word - although unlikely, it could be that the word *catamaran* is not in the lexicon that associates words with tags, and the POS tagger has to make its guess solely on what POS is most likely to appear before a noun, rather than through assessment of lexical probabilities (which would indicate that typically, *catamaran* is a noun rather than an adjective). The sort of POS that occur before nouns are likely to be adjectives, which would explain why the POS tagger erroneously classifies *catamaran* as an adjective.

Similarly, it could be that the POS tagger classifies the prenominal adjective *trial* as a noun because lexically, the word *trial* is most likely to be a noun rather than an adjective. Despite the fact that two nouns (*trial* and *offer*) are not likely to occur sequentially (which would lead the POS tagger to classify *trial* as an adjective), it might be that the probability of *trial* being a noun (based on its probabilities in the tag lexicon) is larger than the probability of it being an adjective (based on the tag sequence likelihoods), which would explain why it was classified as a noun.

(f)

```
S -> NP VP     N -> N PP     Det -> the
PP -> P NP     N -> cat      P -> in
NP -> Det N    N -> box      VP -> snored
N -> Adj N     Adj -> big
```

```
. The . big . cat . in . the . box . snored .
0     1     2     3    4     5     6        7
```

Bottom up Chart

id	left	right	mother	daughter
1	0	1	Det	(the)
2	1	2	Adj	(big)
3	2	3	N	(cat)
4	1	3	N	(1 3)
5	0	3	NP	(0 3)
6	3	4	P	(in)
7	4	5	Det	(the)
8	5	6	N	(box)
9	4	6	NP	(4 6)
10	3	6	PP	(3 6)
11	2	6	N	(2 6)
12	1	6	N	(1 6)
13	0	6	NP	(0 6)
14	6	7	VP	(snored)
15	4	7	S	(4 7)
16	0	7	S	(0 7)

Packed Chart

id	left	right	mother	daughter
1	0	1	Det	(the)
2	1	2	Adj	(big)
3	2	3	N	(cat)
4	1	3	N	(1 3)
5	0	3	NP	(0 3)
6	3	4	P	(in)
7	4	5	Det	(the)
8	5	6	N	(box)
9	4	6	NP	(4 6)
10	3	6	PP	(3 6)
11	2	6	N	(2 6)
12	0	6	NP	(0 6)
13	6	7	VP	(snored)
14	0	7	S	(0 7)

What is meant by the term overgeneration? Why might overgeneration be a problem when using a grammar for parsing grammatical text?

In sentence parsing, it can happen that a parser will generate a large amount of varying structures, a large portion being invalid. This is a result of certain grammar rules permitting the presence of strings that are not meant to go together. These invalid structures then have to be filtered out by application of a set of linguistic principles. It can be a problem as it can lead to performance bottlenecks insofar that many redundant structures are first generated, and then binned. It can also give rise to unwanted ambiguity, generating structures which never occur in reality.