# Natural Language Processing 1

## Lecture 2: Language models and part-of-speech tagging

Katia Shutova

ILLC
University of Amsterdam

28 October 2020

# Outline.

# Part of speech tagging

They can fish.

- They_pronoun can_modal fish_verb.
  ('can' meaning 'are able to')
- They_pronoun can_verb fish_plural-noun.
  ('can' meaning 'put into cans')

Ambiguity

*can*: modal verb, verb, singular noun
*fish*: verb, singular noun, plural noun

# Tagset (CLAWS 5)

tagset: standardized codes for fine-grained parts of speech.
CLAWS 5: over 60 tags, including:

| | | | |
|---|---|---|---|
| NN1 | singular noun | NN2 | plural noun |
| PNP | personal pronoun | VM0 | modal auxiliary verb |
| VVB | base form of verb | VVI | infinitive form of verb |

- They_PNP can_VM0 fish_VVI ._PUN
- They_PNP can_VVB fish_NN2 ._PUN

# POS tagging: Why do we care?

- First step towards syntactic analysis (which in turn, is often useful for semantic analysis).
- Simpler models and often faster than full syntactic parsing, but sometimes enough to be useful
  - POS tags can be useful features in e.g. text classification, authorship identification, etc.
  - Useful for applications such as text to speech synthesis: “it is time to wind the clock up” versus “the wind was strong”

# Extent of POS Ambiguity

The Brown corpus (1,000,000 word tokens) has 39,440 different word types.

- 35340 have only 1 POS tag anywhere in corpus (89.6%)
- 4100 (10.4%) have 2 to 7 POS tags

So why does just 10.4% POS-tag ambiguity by word type lead to difficulty?

# Extent of POS Ambiguity

The Brown corpus (1,000,000 word tokens) has 39,440 different word types.

- 35340 have only 1 POS tag anywhere in corpus (89.6%)
- 4100 (10.4%) have 2 to 7 POS tags

So why does just 10.4% POS-tag ambiguity by word type lead to difficulty?

Many high-frequency words have more than one POS tag.
In fact, around 50% of the word tokens are ambiguous.

# Word Frequencies in Different languages

Ambiguity by part-of-speech tags:

| Language | Type-ambiguity | Token-ambiguity |
|---|---|---|
| English | 13.2% | 56.2% |
| Greek | <1% | 19.14% |
| Japanese | 7.6% | 50.2% |
| Czech | <1% | 14.5% |
| Turkish | 2.5% | 35.2% |

# Some tagging strategies

- One simple strategy: just assign to each word its most common tag. (Call this Uni-gram tagging)
- Surprisingly, even this crude approach typically gives around 90% accuracy. (State-of-the-art (English) is 97 - 98%).
- Can we do better?

# Part of speech tagging using Hidden Markov Models (HMM)

1. Start with untagged text.
2. Assign all possible tags to each word in the text on the basis of a lexicon that associates words and tags.
3. Find the most probable sequence (or n-best sequences) of tags, based on probabilities from the training data.
   - lexical probability: e.g., is *can* most likely to be VM0, VVB, VVI or NN1?
   - and tag sequence probabilities: e.g., is VM0 or NN1 more likely after PNP?

# Assigning probabilities

Estimate tag sequence: *n* tags with the maximum probability, given *n* words:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}}\, P(t_1^n | w_1^n)$$

a tag sequence, which contains n tags

a word sequence, which contains n words

estimated tag sequence: means the tag seq which has the highest prob

lexical prob. = likelihood = prob of a word sequence given a tag sequence

By Bayes theorem:

$$\text{posterior}\quad P(t_1^n | w_1^n) = \frac{P(w_1^n | t_1^n) P(t_1^n)}{P(w_1^n)}$$

likelihood prior evid

the most probable tag sequence, given the tag of previous word. how likely is the next tag, given the tags that we have seen

but $P(w_1^n)$ is constant:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}}\, P(w_1^n | t_1^n) P(t_1^n)$$

output is: product of lexical prob and tag prob. we want to compare the possibility of all tag sequence for the same word sequence.

lexical prob: prob of word given the tag

prob of tag sequence. how likeli is the next tag, given the tag we have seen perviously. in n-gram, prior is a conditional prob

# Bigrams

Bigram assumption: probability of a tag depends on previous tag, hence product of bigrams:

t^n: a sequence of n tags

$$P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1})$$

the prob of next tag, given one previous tag. ust one previous tag. because we choose bigrams

Probability of word estimated on basis of its tag alone:

the prob of word seq, given tag seq.
w^n : a word seq which has n words
t^n: a tag seq which has n tags

this is break down to:

右上角有n: sequence
右上角无n: single word or single

$$P(w_1^n|t_1^n) \approx \prod_{i=1}^{n} P(w_i|t_i)$$

prob of a single word, given a single tag

p(wi|ti): the prob that a particular word has a particular tag
= the prob that a particular tag generates a particular word.
p(wi|ti) is indep from other words and other tags in the sentence. this is purely lexical prob. because it is just about the tag and word. lexical prob don't use previous things to estimate the current things

loop through all word.
For each words, we compute the product of these 2 prob.
argmax
{ [ p(w1|t1)p(t1|t0) ]
* [ p(w2|t2)p(t2|t1) ]
* [ p(w3|t3)p(t3|t2) ] }

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1})$$

lexical prob

tag sequence prob

# Example

this gives a very low probablilty. because it is rare to see 人称代词+名词的语法结构 所以this term makes the whole multiplied product be discarded

Tagging: *they fish* (ignoring punctuation)

Assume PNP is the only tag for *they*, and that *fish* could be NN2 or VVB.

fish can be a none or a verb

人称代词pronoun

Then the estimate for PNP NN2 will be:

$$P(\text{they}|\text{PNP})\ P(\text{NN2}|\text{PNP})\ P(\text{fish}|\text{NN2}) \cdot P(t_1 \mid t_0) = 1$$

and for PNP VVB:

$$P(\text{they}|\text{PNP})\ P(\text{VVB}|\text{PNP})\ P(\text{fish}|\text{VVB}) \cdot P(t_1 \mid t_0) = 1$$

这是一组 product of lexicon prob and tag seq prob

# Training the POS tagger

They_PNP used_VVD to_TO0 can_VVI fish_NN2 in_PRP those_DT0 towns_NN2 ._PUN But_CJC now_AV0 few_DT0 people_NN2 fish_VVB in_PRP these_DT0 areas_NN2 ._PUN

先count how many times this occurs, 再compute prob

| sequence | count | bigram probability |
|---|---|---|
| NN2 | 4 | |
| NN2 PRP | 1 | 0.25 |
| NN2 PUN | 2 | 0.5 |
| NN2 VVB | 1 | 0.25 |

NN2 appears 4 times in total, in the above corpus

先名词，后介词，出现了一次

先名词，后断句，出现了2次

Also lexicon: fish NN2 VVB

# Training the POS tagger

```
They_PNP used_VVD to_TO0 can_VVI fish_NN2 in_PRP
those_DT0 towns_NN2 ._PUN But_CJC now_AV0 few_DT0
people_NN2 fish_VVB in_PRP these_DT0 areas_NN2
._PUN
```

| sequence | count | bigram probability |
|---|---|---|
| NN2 | 4 | |
| NN2 PRP | 1 | 0.25 |
| NN2 PUN | 2 | 0.5 |
| NN2 VVB | 1 | 0.25 |

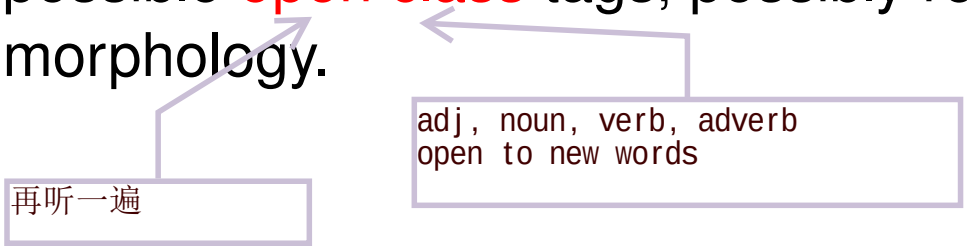
Also lexicon: fish NN2 VVB

根据以上表格，可以算出对于每一个word，它拥有哪些tag options，以及 how likely this word belongs to tag1. how likely this word belongs to tag2.
eg. we record 2 prob for fish, in our lexicon:
prob that fish belongs to noun
prob that fish belongs to verb

then we use this lexicon to annotate new text

# Applying in practice

- Maximise the overall tag sequence probability
- Actual systems use trigrams — smoothing and backoff are critical.
- Unseen words: these are not in the lexicon, so use all possible open class tags, possibly restricted by morphology.



# Evaluation of POS tagging

- percentage of correct tags, i.e. accuracy
- one tag per word (some systems give multiple tags when uncertain)
- accuracy over 97% for English (but note punctuation is unambiguous)
- baseline of taking the most common tag gives 90% accuracy

baseline: is a simple technique which we know it definitely works
we want to compare our model with the baseline

# Other tagging or sequence labelling tasks

- Named entity recognition: e.g., label words as belonging to persons, organizations, locations, or none of the above:

  *Barack/PER Obama/PER spoke/NON from/NON the/NON White/LOC House/LOC today/NON ./NON*

- Information field segmentation: Given specific type of text (e.g. classified advert), identify which words belong to which fields (e.g. price/ size/ location)

  *3BR/SIZE flat/TYPE in/NON Bruntsfield/LOC ,/NON near/LOC main/LOC roads/LOC ./NON Bright/FEAT ,/NON well/FEAT maintained/FEAT ...*

Correct tags depend on the sequence of words.

# Acknowledgement

*Some slides were adapted from Ann Copestake, Dan Jurafsky and Tejaswini Deoskar*