# Outline of today's lecture

Lecture 3: Prediction and part-of-speech tagging
- Corpora in NLP
- Word prediction
- Part-of-speech (POS) tagging
- Evaluation in general, evaluation of POS tagging

First of three lectures that concern syntax (i.e., how words fit together). This lecture: 'shallow' syntax: word sequences and POS tags. Next lectures: more detailed syntactic structures.

# Corpora

- corpus: text that has been collected for some purpose.
- balanced corpus: texts representing different genres
  genre is a type of text (vs domain)
- tagged corpus: a corpus annotated with POS tags
- treebank: a corpus annotated with parse trees
- specialist corpora — e.g., collected to train or evaluate particular applications
  - Movie reviews for sentiment classification
  - Data collected from simulation of a dialogue system

# Uses of prediction

- unsupervised training for various models (esp. neural networks, lecture 9).
- language modelling for broad-coverage speech recognition to disambiguate results from signal processing: e.g., using n-grams or (recently) LSTMs.
- word prediction for communication aids:
  e.g., to help enter text that's input to a synthesiser
- text entry on mobile phones and similar devices
- spelling correction, text segmentation
- estimation of entropy

# bigrams (n-gram with N=2)

A probability is assigned to a word based on the previous word:

$$P(w_n|w_{n-1})$$

where $w_n$ is the nth word in a sentence.
Probability of a sequence of words (assuming independence):

$$P(W_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1})$$

# bigrams: probability estimation

Probability is estimated from counts in a training corpus:

$$\frac{C(w_{n-1}w_n)}{\sum_w C(w_{n-1}w)} \approx \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

i.e. count of a particular bigram in the corpus divided by the count of all bigrams starting with the prior word.

⟨s⟩ good morning ⟨/s⟩ ⟨s⟩ good afternoon ⟨/s⟩ ⟨s⟩ good afternoon ⟨/s⟩ ⟨s⟩ it is very good ⟨/s⟩ ⟨s⟩ it is good ⟨/s⟩

| sequence | count | bigram probability |
|---|---|---|
| ⟨s⟩ | 5 | |
| ⟨s⟩ good | 3 | .6 |
| ⟨s⟩ it | 2 | .4 |
| good | 5 | |
| good morning | 1 | .2 |
| good afternoon | 2 | .4 |
| good ⟨/s⟩ | 2 | .4 |
| ⟨/s⟩ | 5 | |
| ⟨/s⟩ ⟨s⟩ | 4 | 1 |

# Sentence probabilities

Probability of ⟨s⟩ it is good afternoon ⟨/s⟩ is estimated as:
$P(\text{it}|\langle\text{s}\rangle)P(\text{is}|\text{it})P(\text{good}|\text{is})P(\text{afternoon}|\text{good})P(\langle/\text{s}\rangle|\text{afternoon})$
$= .4 \times 1 \times .5 \times .4 \times 1 = .08$

What about the probability of ⟨s⟩ very good ⟨/s⟩ ?
$P(\text{very}|\langle\text{s}\rangle)$?

# Sentence probabilities

Probability of $\langle s \rangle$ it is good afternoon $\langle /s \rangle$ is estimated as:
$P(\text{it}|\langle s \rangle)P(\text{is}|\text{it})P(\text{good}|\text{is})P(\text{afternoon}|\text{good})P(\langle /s \rangle|\text{afternoon})$
$= .4 \times 1 \times .5 \times .4 \times 1 = .08$

What about the probability of $\langle s \rangle$ very good $\langle /s \rangle$ ?
$P(\text{very}|\langle s \rangle)$?

# Sentence probabilities

Probability of $\langle s \rangle$ it is good afternoon $\langle /s \rangle$ is estimated as:
$P(\text{it}|\langle s \rangle)P(\text{is}|\text{it})P(\text{good}|\text{is})P(\text{afternoon}|\text{good})P(\langle /s \rangle|\text{afternoon})$
$= .4 \times 1 \times .5 \times .4 \times 1 = .08$

What about the probability of $\langle s \rangle$ very good $\langle /s \rangle$ ?
$P(\text{very}|\langle s \rangle)$?

# Sentence probabilities

Problems because of sparse data:

- smoothing: distribute 'extra' probability between rare and unseen events (e.g., add-one smoothing)
- backoff: approximate unseen probabilities by a more general probability, e.g. unigrams

cf Chomsky: *Colorless green ideas sleep furiously*
smoothing means unseen phrases have a non-zero probability estimate.

# Sentence probabilities

Problems because of sparse data:

- smoothing: distribute ‘extra’ probability between rare and unseen events (e.g., add-one smoothing)
- backoff: approximate unseen probabilities by a more general probability, e.g. unigrams

cf Chomsky: *Colorless green ideas sleep furiously*
smoothing means unseen phrases have a non-zero probability estimate.

# Practical application

- Word prediction: guess the word from initial letters. User confirms each word, so we predict on the basis of individual bigrams consistent with letters.
- Speech recognition: given an input which is a lattice of possible words, we find the sequence with maximum likelihood.
  Implemented efficiently using dynamic programming (Viterbi algorithm).

# 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
- They_PNP can_VM0 fish_NN2 ._PUN no full parse
- etc

# Why POS tag?

Coarse-grained syntax / word sense disambiguation: fast, so applicable to very large corpora.

- Some linguistic research and lexicography: e.g., how often is *tango* used as a verb? *dog*?
- Named entity recognition and similar tasks (finite state patterns over POS tagged data).
- Features for machine learning e.g., sentiment classification. (e.g., *stink_V* vs *stink_N*).
- Fast preliminary processing for full parsing: provide guesses at unknown words, cut down search space.

# Stochastic 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:

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

By Bayes theorem:

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

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

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

# Bigrams

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

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

Probability of word estimated on basis of its tag alone:

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

Hence:

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

# Example

Tagging: *they fish* (ignoring punctuation)
Assume PNP is the only tag for *they*, and that *fish* could be NN2 or VVB.
Then the estimate for PNP NN2 will be:

$$P(\text{they}|\text{PNP})\ P(\text{NN2}|\text{PNP})\ P(\text{fish}|\text{NN2})$$

and for PNP VVB:

$$P(\text{they}|\text{PNP})\ P(\text{VVB}|\text{PNP})\ P(\text{fish}|\text{VVB})$$

# Training stochastic POS tagging

```
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

# Training stochastic POS tagging

```
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

# Training stochastic POS tagging

```
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

# Assigning probabilities, more details

- Maximise the overall tag sequence probability — e.g., use Viterbi.
- 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
- one tag per word (some systems give multiple tags when uncertain)
- over 95% for English on normal corpora (but note punctuation is unambiguous)
- performance plateau about 97% on most commonly used test set for English
- baseline of taking the most common tag gives 90% accuracy
- different tagsets give slightly different results: utility of tag to end users vs predictive power

# Evaluation in general

- Training data and test data Test data must be kept unseen, often 90% training and 10% test data.
- Baseline
- Ceiling Human performance on the task, where the ceiling is the percentage agreement found between two annotators (interannotator agreement)
- Error analysis Error rates are nearly always unevenly distributed.
- Reproducibility

# Representative corpora and data sparsity

- test corpora have to be representative of the actual application
- POS tagging and similar techniques are not always very robust to differences in genre
- balanced corpora may be better, but still don't cover all text types
- communication aids: extreme difficulty in obtaining data, text corpora don't give good prediction for real data