Today’s Lecture

- Recap
- Part-of-speech tagging
- Hidden Markov Model (HMM)
Recap – Models

\[ f : x \rightarrow y \]

- Non-probabilistic: \( f \)
- Discriminative: \( P(y|x) \)
- Generative: \( P(x, y) \)
Recap – Naive Bayes

- Fixed vocabulary
- Feature vectors $x_i$
  - Binary (Bernoulli NB)
  - Bag of words (Multinomial NB)
- Parameters $P(x_i|y), P(y)$
- Train using observed counts
Vocabulary Size

**Zipf’s Law**: word frequency follows a power law distribution; many words only appear once (long tail)

**Heaps’ Law**: no matter how much data we observe (tokens), we will never see all words (types)
Zipf’s Law: word frequency follows a power law distribution; many words only appear once (long tail)

Heaps’ Law: no matter how much data we observe (tokens), we will never see all words (types)

⇒ Some words unseen at test time
Part-of-Speech Tagging

They can fish.

PNP VM0 VVI PUN

CLAWS-5 tagset includes:

NN1 singular noun VVB verb, base form
NN2 plural noun VVI verb, infinitive
PNP personal pronoun VM0 verb, modal
PUN punctuation VVZ verb, 3rd pers. sg.
Part-of-Speech Tagging

They can fish.

CLAWS-5 tagset includes:

- **NN1**: singular noun
- **NN2**: plural noun
- **PNP**: personal pronoun
- **PUN**: punctuation
- **VV1**: verb, infinitive
- **VVZ**: verb, 3rd pers. sg.
- **VVB**: verb, base form
- **VM0**: verb, modal
Part-of-Speech Tagging

They can fish .

PNP VM0 VVI PUN
PNP VVB NN2 PUN
PNP VM0 NN2 PUN

CLAWS-5 tagset includes:

NN1 singular noun
NN2 plural noun
PNP personal pronoun
PUN punctuation

VVB verb, base form
VVI verb, infinitive
VM0 verb, modal
VVZ verb, 3rd pers. sg.
Part-of-Speech Lexicon Fragment

d they PNP
  can VM0, NN1, VVB, VVI
fish NN2, NN1, VVB, VVI
Part-of-Speech Lexicon Fragment

they  PNP

can   VM0, NN1, VVB, VVI

fish  NN2, NN1, VVB, VVI

- Could be hand-written
- ML: aim to learn from data
Why Do Part-of-Speech Tagging?

- Not often considered until 1990s
Why Do Part-of-Speech Tagging?

- Not often considered until 1990s
- Easier than full parsing
Why Do Part-of-Speech Tagging?

- Not often considered until 1990s
- Easier than full parsing
- For applications:
  - Reduce search space for unknown words
  - Input features for other tasks
Why Do Part-of-Speech Tagging?

- Not often considered until 1990s
- Easier than full parsing
- For applications:
  - Reduce search space for unknown words
  - Input features for other tasks
- For linguistics:
  - Lexicography
  - Corpus linguistics
Defining The Task

- Which language? (dialect?)
- Tagset? Syntactic analysis?
- Genre? Domain?
Defining The Task

- Errors in the input?
  They walked into into the room

- Errors in the annotations?

- Rare words / rare usages of words?
Defining The Task

- Sequence labelling
  - Input: sequence
  - Output: sequence of same length

- Usually supervised
Data

- Limited training data
  - Requires trained annotators
  - Annotation guidelines are lengthy
- High inter-annotator agreement
Hidden Markov Model

\[
\underset{t_1 \cdots t_n}{\text{argmax}} \ P(t_1 \cdots t_n | w_1 \cdots w_n)
\]
Hidden Markov Model

\[
\begin{align*}
\text{argmax } & P(t_1 \cdots t_n | w_1 \cdots w_n) \\
& \approx \text{argmax } P(t_1 \cdots t_n) P(w_1 \cdots w_n | t_1 \cdots t_n)
\end{align*}
\]
Hidden Markov Model

\[
\text{argmax}_{t_1 \cdots t_n} P(t_1 \cdots t_n | w_1 \cdots w_n) \\
= \text{argmax}_{t_1 \cdots t_n} P(t_1 \cdots t_n) P(w_1 \cdots w_n | t_1 \cdots t_n) \\
\approx \text{argmax}_{t_1 \cdots t_n} \prod_{i=1}^{n} P(t_i | t_{i-1}) P(w_i | t_i)
\]
Hidden Markov Model
Parameters:

- \( P(t_i|t_{i-1}) \) – transition probabilities
- \( P(w_i|t_i) \) – emission probabilities
Hidden Markov Model

Parameters:
- \( P(t_i|t_{i-1}) \) – transition probabilities
- \( P(w_i|t_i) \) – emission probabilities

Train using observed counts

Smoothing hyperparameters
Backoff

For higher $n$-grams, can use backoff:

$$
\hat{P}(t_i|t_{i-1}, t_{i-2}) = \lambda P_{\text{trigram}}(t_i|t_{i-1}, t_{i-2}) \\
+ (1 - \lambda) P_{\text{bigram}}(t_i|t_{i-1})
$$

$P_{\text{trigram}}, P_{\text{bigram}}$ calculated using counts
Inference

- An HMM learns $P(t_1 \cdots t_n, w_1 \cdots w_n)$

- Dynamic programming:
  - Most likely sequence $\rightarrow$ Viterbi Algorithm
  - Most likely tag for each word $\rightarrow$ Inside-Outside Algorithm
Unknown Words

- Frequency of open-class tags
- Morphology (e.g. “-ing”)
- Capitalisation (e.g. “Bill” vs. “bill”)
Discriminative POS-Tagging

- Conditional Random Fields
- Recurrent Neural Networks (details in future lecture)
State of the Art

- Plank et al. (2016), in course readings
- Performance close to ceiling
- Return to question: what is the task?