# NLP1 2023/24

Sequence Labelling

Lecturer: Wilker Aziz

(week 2, lecture a)

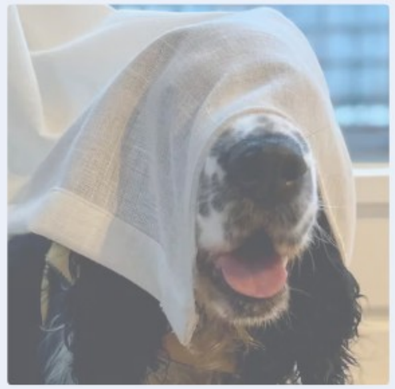
# Where are we at?

- → What makes NLP hard
- → Text classification
- → Language modelling
- → **Sequence labelling**

# Self-study: logistic CPDs (theory and example)

Yay

25



Nay

22

# Ask me anything about logistic CPDs

4 questions
7 upvotes

# Outline

- **Word classes**
- Hidden Markov model
- Evaluation
- Sequence labelling
- Conditional random fields

# We can organise words into classes

- semantic criteria: what does the word refer to?
  - nouns often refer to ‘people’, ‘places’ or ‘things’
- formal criteria: what form does the word have?
  - -ly makes an adverb out of an adjective
  - -tion makes a noun out of a verb
- distributional criteria: in what contexts can the word occur?
  - adjectives precede nouns

## But why do that?

10 responses

| | | |
|---|---|---|
| Reduce the number of dimensions | What is your native language? | Lower dimensionality |
| fixes ambiguity | to generalize prediction | help improve vocabulary |
| Smaller representation | Conditional probabilities. Feature extraction | Vibes |

## But why do that?

10 responses

We can use these classes as features

# But why?

Remember one of the limitations of our tabular CPDs? We treated words as if they were completely unrelated to one another, word classes capture some aspects of word relatedness.

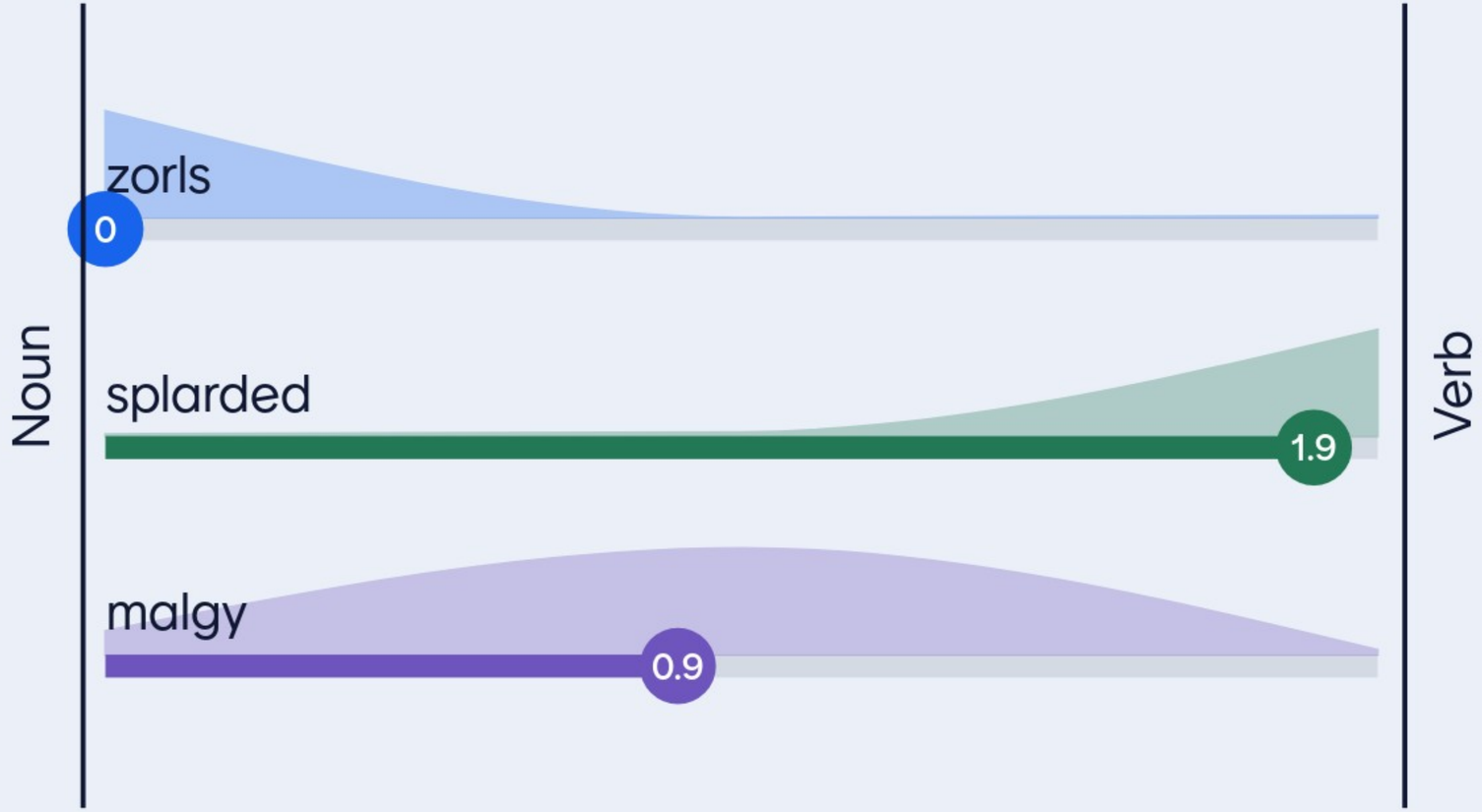
| | Semantically | Formally | Distributionally |
|---|---|---|---|
| Nouns | refer to things, concepts | -ness, -tion, -ity, -ance | After determiners, possessives |
| Verbs | refer to actions, states | -ate, -ize | infinitives: to jump, to learn |
| Adjectives | properties of nouns | -al, -ble | appear before nouns |
| Adverbs | properties of actions | -ly | next to verbs, beginning of sentence |

Examples of criteria

# More on motivation

Word classes enable a form of delexicalised natural language processing in which we can learn about patterns that are common to all words that share a given property (e.g., in English, a pronoun is typically followed by a verb).

# Those zorls you splarded were malgy



# How many classes are there?

This is very much language dependent.

The English Brown corpus has 87, the Penn Treebank has 45.

Universal POS tags are a simplifying tags aimed at cross-lingual compatibility (it maps variants of a base class to that base class, e.g.,
VBD, VBN, VB, VBG, VBP → VERB)

# Universal parts-of-speech (POS)

ADJ (adjectives)
ADP (prepositions and postpositions)
ADV (adverbs)
CONJ (conjunctions)
DET (determiners and articles)
NOUN (nouns)
NUM (numerals)
PRON (pronouns)
PRT (particles)
PUNCT (punctuation marks)
VERB (verbs)
X (anything else, such as abbreviations or foreign words)

# Example (PennTreebank-style)

The/**DT** grand/**JJ** jury/**NN**
commented/**VBD** on/**IN** a/**DT**
number/**NN** of/**IN** other/**JJ** topics/**NNS**
./.

There/**EX** was/**VBD** still/**JJ**
lemonade/**NN** in/**IN** the/**DT** bottle/**NN** ./.

# Goals for this class: to learn

- how to *model POS-tagged data*
- how to *model text using word classes*
- a general approach to *sequence labelling problems*

# Outline

- Word classes
- **Hidden Markov model**
- Evaluation
- Sequence labelling
- Conditional random fields

# POS-tagged data

We will prescribe a **joint distribution** over the space of **texts annotated with their POS tags**.

That is, we will be learning to assign probability to sequence pairs of the kind $(w_{1:l}, c_{1:l})$
where $w_{1:l}$ is a word sequence and $c_{1:l}$ is the corresponding POS tag sequence.

Example: (<a, nice, dog>, <DT, JJ, NN>)

# Applications

- Text analysis: annotating text with POS tags (e.g., input to other tools)
- Language modelling: address some limitations of NGram LMs
- Also, the ideas we develop now will prove useful in many labelling tasks

# Formalisation

$W$ is a random *word*. An outcome $w$ is a symbol in a vocabulary $\mathcal{W}$ of size $V$.
$C$ is a random *POS tag*. An outcome $c$ is a symbol in the tagset $\mathcal{C}$ of size $K$.

$X = \langle W_1, \ldots, W_L \rangle$ is a random *sequence*. An outcome $w_{1:l}$ is a sequence of $l$ words from $\mathcal{W}$.
$Y = \langle C_1, \ldots, C_L \rangle$ is a random *sequence*. An outcome $c_{1:l}$ is a sequence of $l$ tags from $\mathcal{C}$.

# Statistical task

Design a mechanism to assign probability $P_{XY}\left(w_{1:l}, c_{1:l}\right)$ to **any** outcome $\left(w_{1:l}, c_{1:l}\right) \in \mathcal{W}^* \times \mathcal{C}^*$ .

Estimate the parameters of this mechanism from data (i.e., text annotated with POS tags).

# NLP tasks

Predict a POS tag sequence for a given text.
For example via mode-seeking search:

$$\arg\max_{c_{1:l} \in \mathcal{C}^l} P_{Y|X}(c_{1:l} | w_{1:l})$$

Assign probability to text that is **not** annotated with POS tags via marginalisation:

$$P_X(w_{1:l}) = \sum_{c_1=1}^{K} \cdots \sum_{c_l=1}^{K} P_{XY}(w_{1:l}, c_{1:l})$$

# Challenge

$P_{XY}$ is a distribution over a countably infinite space of sequence pairs.

There is no standard statistical distribution over such a sample space. Hence, we need to develop one.

# Key idea

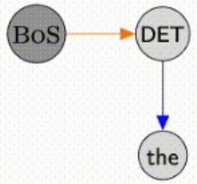
Re-express the probability of a sequence pair using the probabilities of the "steps" needed to generate it.

Design steps such that they have a simple, countably finite sample space.

Joint observations

the/*DET* book/*NOUN* is/*VERB* on/*ADP* the/*DET* table/*NOUN* ./*PUNC*

Generative story




We pad the tag sequence with a BoS symbol. We pad both sequences with a EoS symbol.

HMM factorisation of the probability of a sequence pair in terms of transition and emission probabilities.

# Chain rule for the HMM

$$P_{XY}(w_{1:l}, c_{1:l}) = \prod_{i=1}^{l} \underbrace{P_{C|C_{\text{prev}}}(c_i|c_{i-1})}_{\text{transition}} \underbrace{P_{W|C}(w_i|c_i)}_{\text{emission}}$$

*Hint.* Pad the sequences with a special BOS token (or tag) and a special EOS token (or tag).

To learn a bit about graphical models, see our Introduction to PGMs.

# Generative story

1. Start with $X = \langle W_0 = \mathrm{BOS} \rangle, Y = \langle C_0 = \mathrm{BOS} \rangle$ and set $i = 1$;

2. Condition on the previous class $c_{i-1}$ and draw a class $c_i$ with probability $P_{C|C_{\mathrm{prev}}}(c_i|c_{i-1})$ extending $Y$ with it;

3. Condition on the current class $c_i$ and draw a word $w_i$ with probability $P_{W|C}(w_i|c_i)$ extending $X$ with it;

4. If $w_i$ is a special end-of-sequence symbol (EOS), terminate, else increment $i$ and repeat from (2).

This specifies a **factorisation** of $P_{XY}$ in terms of elementary factors of the kind $P_{C|C_{\mathrm{prev}}}$ and $P_{W|C}$.

# Tabular parameterisation

**Transition distribution**

$$C|C_{\text{prev}} = r \sim \text{Categorical}(\lambda^{(r)}_{1:K})$$

**Emission distribution**

$$W|C = c \sim \text{Categorical}(\theta^{(c)}_{1:V})$$

**Probability mass function (pmf):**

$$p(w_{1:l}, c_{1:l}; \boldsymbol{\theta}, \boldsymbol{\lambda}) =$$

$$\prod_{i=1}^{l} \underbrace{\lambda^{(c_{i-1})}_{c_i}}_{\text{transition}} \times \underbrace{\theta^{(c_i)}_{w_i}}_{\text{emission}}$$

# Tabular parameterisation

**Probability mass function (pmf):**

$$p(w_{1:l}, c_{1:l}; \boldsymbol{\theta}, \boldsymbol{\lambda}) = \prod_{i=1}^{l} \underbrace{\lambda_{c_i}^{(c_{i-1})}}_{\text{transition}} \times \underbrace{\theta_{w_i}^{(c_i)}}_{\text{emission}}$$

**Example:**

$$p(\langle \text{a}, \text{nice}, \text{dog} \rangle, \langle DT, JJ, NN \rangle; \boldsymbol{\theta}, \boldsymbol{\lambda}) = \lambda_{\text{DT}}^{(\text{BOS})} \theta_{\text{a}}^{(\text{DT})} \lambda_{\text{JJ}}^{(\text{DT})} \theta_{\text{nice}}^{(\text{JJ})} \lambda_{\text{NN}}^{(\text{JJ})} \theta_{\text{dog}}^{(\text{NN})} \lambda_{\text{EOS}}^{(\text{NN})} \theta_{\text{EOS}}^{(\text{EOS})}$$

## Express the MLE for transition probability $r \rightarrow c$

11 responses

proportion of second tag preceded by first tag in training data

P(c{*step*}|c{step-1})

transitions from c / count of c

argmax_lambda lambda^r

Number of tags c following r divided by the count of all r tags

Count(c,r)/count(c)

Bayes rule

p(r, c) / p(c)

count( r → c ) / count( c )

## Express the MLE for transition probability $r \rightarrow c$

11 responses

P(r|c)

Count r,c/ sum_j Count r,j

# Parameter Estimation: Transitions

For each possible previous tag (any $r \in \mathcal{C} \cup \{\text{BOS}\}$), we have a transition cpd over $K$ possible tags:

$$C|C_{\text{prev}} = r \sim \text{Categorical}(\lambda_{1:K}^{(r)})$$

Given a dataset of observed texts annotated with POS, the maximum likelihood estimate of the conditional probability $\lambda_c^{(r)}$ of generating a tag $c$ right after having generated a tag $r$ is:

$$\begin{aligned}\lambda_c^{(r)} &= \frac{\text{count}_{C_{\text{prev}}C}(r, c)}{\sum_{k=1}^{K} \text{count}_{C_{\text{prev}}C}(r, k)} \\ &= \frac{\text{count}_{C_{\text{prev}}C}(r, c)}{\text{count}_{C_{\text{prev}}}(r)}\end{aligned}$$

## Express the MLE for emission probability $c \rightarrow w$

8 responses

| | | |
|---|---|---|
| count(c,w)/count(c) | Count(c,w)/count(c) | P(w\|c) |
| Count( c, w) / count(c) | count(w, c) / count(c) | P( w\|c) = count (c, w)/count(c) |
| count( c, w ) / count( c ) | count(c,w)/count(c) | |

# Parameter Estimation: Emissions

For each possible tag $c \in \mathcal{C}$ (the EOS tag is assumed to be part of the tagset), we have an emission cpd over $V$ tokens:

$$W|C = c \sim \text{Categorical}(\theta^{(c)}_{1:V})$$

Given a dataset of observed texts annotated with POS, the maximum likelihood estimate of the conditional probability $\theta^{(c)}_w$ of generating word $w$ from tag $c$ is:

$$\begin{aligned}\theta^{(c)}_w &= \frac{\text{count}_{CW}(c, w)}{\sum_{o=1}^{V} \text{count}_{CW}(c, o)} \\ &= \frac{\text{count}_{CW}(c, w)}{\text{count}_C(c)}\end{aligned}$$

# Data Sparsity

It's still possible that this model suffers from data sparsity (e.g., unseen word-tag pairs or unseen tag-tag pairs), but much less so than an NGram LM: contextual information is only available through the POS tag of the previous position ($K$ possible outcomes, instead of $V^{n-1}$ outcomes).

## Limitations
16 responses

Samples will be nonsense that's vaguely grammatical

POS tags contain no semantic information

knowledge about previous words

wrong grammar

We will get gibberish sentences.

Attention is all you need. This model has no attention

Grammatically correct gibberish

gobaldygoo

you'll get stuff like "Colorless green ideas sleep furiously"

## Limitations
16 responses

No semantic context information ( only grammatical)

bobbledygook

Can you use longer Markov sequences? e.g. condition on the previous three POSs

no way to see further in the past then 1 step

the book is on the table, the hable is on the book

always same sentences?

Modern ARt

# Questions about HMMs?

4 questions
7 upvotes

# Outline

- Word classes
- Hidden Markov model
- **Evaluation**
- Sequence labelling
- Conditional random fields

# Tagging Performance

Predict a POS tag sequence for novel text. For example via mode-seeking search:

$$\hat{c}_{1:l} = \arg\max_{c_{1:l} \in \mathcal{C}^l} P_{Y|X}(c_{1:l}|w_{1:l})$$

Compare predicted $\hat{c}_{1:l}$ to human-annotated $c^{\star}_{1:l}$ step by step: assess the rate at which the $i$th prediction matches the $i$th target (**accuracy**).

# Most Probable Tag Sequence

We look for the posterior mode:

$$\arg\max_{c_{1:l} \in \mathcal{C}^l} P_{Y|X}(c_{1:l}|w_{1:l})$$

Definition of conditional probability:

$$= \arg\max_{c_{1:l} \in \mathcal{C}^l} \frac{P_{XY}(w_{1:l}, c_{1:l})}{P_X(w_{1:l})}$$

The argmax is constant wrt $P_X(w_{1:l})$:

$$= \arg\max_{c_{1:l} \in \mathcal{C}^l} P_{XY}(w_{1:l}, c_{1:l})$$

HMM factorisation:

$$= \arg\max_{c_{1:l} \in \mathcal{C}^l} \prod_{i=1}^{l} P_{C|C_{\text{prev}}}(c_i|c_{i-1}) P_{W|C}(w_i|c_i)$$

Categorical pmf:

$$= \arg\max_{c_{1:l} \in \mathcal{C}^l} \prod_{i=1}^{l} \lambda_{c_i}^{(c_{i-1})} \times \theta_{w_i}^{(c_i)}$$

Monotonicity of $\log$ and numerical convenience:

$$= \arg\max_{c_{1:l} \in \mathcal{C}^l} \sum_{i=1}^{l} \log \lambda_{c_i}^{(c_{i-1})} + \log \theta_{w_i}^{(c_i)}$$

Example:

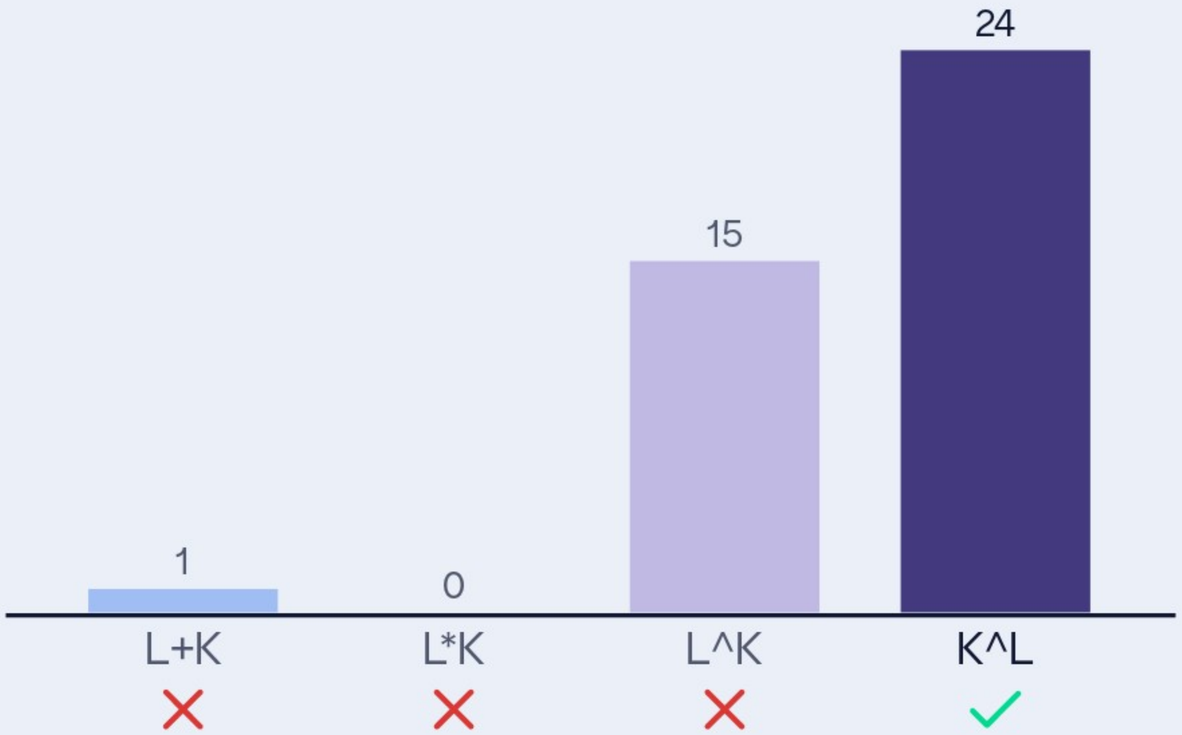
observation $w_{1:3} \circ \langle \mathrm{EOS} \rangle$

tagset $\{\mathsf{A}, \mathsf{B}\}$

| $C_0$ | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $\prod_{i=1}^{l} \lambda_{c_i}^{(c_{i-1})} \times \theta_{w_i}^{(c_i)}$ |
|---|---|---|---|---|---|

Brute force: enumerate sequences, score, sort, pick best

# With $K$ tags in the tagset, how many POS tag sequences of length L are there?



| $C_1$ | $C_2$ | $C_3$ | $C_4$ | $\prod_{i=1}^{l} \lambda_{c_i}^{(c_{i-1})} \times \theta_{w_i}^{(c_i)}$ |
|---|---|---|---|---|
| A | A | A | EoS | $\lambda_{\mathsf{A}}^{(\mathsf{BoS})} \times \theta_{w_1}^{(\mathsf{A})} \times \lambda_{\mathsf{A}}^{(\mathsf{A})} \times \theta_{w_2}^{(\mathsf{A})} \times \lambda_{\mathsf{A}}^{(\mathsf{A})} \times \theta_{w_3}^{(\mathsf{A})} \times \lambda_{\mathsf{EoS}}^{(\mathsf{A})} \times \theta_{\mathrm{EoS}}^{(\mathsf{EoS})}$ |
| A | A | B | EoS | $\lambda_{\mathsf{A}}^{(\mathsf{BoS})} \times \theta_{w_1}^{(\mathsf{A})} \times \lambda_{\mathsf{A}}^{(\mathsf{A})} \times \theta_{w_2}^{(\mathsf{A})} \times \lambda_{\mathsf{B}}^{(\mathsf{A})} \times \theta_{w_3}^{(\mathsf{B})} \times \lambda_{\mathsf{EoS}}^{(\mathsf{B})} \times \theta_{\mathrm{EoS}}^{(\mathsf{EoS})}$ |
| A | B | A | EoS | $\lambda_{\mathsf{A}}^{(\mathsf{BoS})} \times \theta_{w_1}^{(\mathsf{A})} \times \lambda_{\mathsf{B}}^{(\mathsf{A})} \times \theta_{w_2}^{(\mathsf{B})} \times \lambda_{\mathsf{A}}^{(\mathsf{B})} \times \theta_{w_3}^{(\mathsf{A})} \times \lambda_{\mathsf{EoS}}^{(\mathsf{A})} \times \theta_{\mathrm{EoS}}^{(\mathsf{EoS})}$ |

# Dynamic programming for the rescue

Exact enumeration is intractable, but, as it turns out, it's unnecessary.

Because of the conditional independences in the HMM, changing the POS tag of position $i$ can only affect one emission probability ($\underline{C_i} \rightarrow w_i$) and two transition probabilities ($C_{i-1} \rightarrow \underline{C_i}$ and $\underline{C_i} \rightarrow C_{i+1}$). This allows us to solve the problem incrementally from left to right in time $\mathcal{O}(L \times K^2)$.

# Viterbi Algorithm

For a **given** text $w_{1:l}$,
$\alpha(i, j)$ is the maximum probability of a sequence ending in $(C_i = j, W_i = w_i)$:

$$\alpha(i,j) = \begin{cases} \lambda_j^{(\mathrm{BoS})} \theta_{w_i}^{(j)} & \text{if } i = 1 \\ \max\limits_{r \in \mathcal{C}} \ \alpha(i-1, r) \lambda_j^{(r)} \theta_{w_i}^{(j)} & \text{if } i > 1 \end{cases}$$

Then, $\alpha(l+1, \mathrm{EoS})$ is the mode probability.

Store the $\arg\max$ at each step to obtain the POS tag sequence with maximum probability.

Watch the video I prepared for you.

# LM Performance

Use the HMM to assign probability to observed text $w_{1:l}$

$$P_X(w_{1:l}) = \sum_{c_1 \in \mathcal{C}} \cdots \sum_{c_l \in \mathcal{C}} P_{XY}(w_{1:l}, c_{1:l})$$

$$= \sum_{c_1 \in \mathcal{C}} \cdots \sum_{c_l \in \mathcal{C}} \prod_{i=1}^{l} P_{C|C_{\text{prev}}}(c_i|c_{i-1}) P_{W|C}(w_i|c_i)$$

$$= \sum_{c_1 \in \mathcal{C}} \cdots \sum_{c_l \in \mathcal{C}} \prod_{i=1}^{l} \lambda_{c_i}^{(c_{i-1})} \times \theta_{w_i}^{(c_i)}$$

Use a heldout dataset and the **marginal** pmf to assess the perplexity of the model.

Example:

observation $w_{1:3} \circ \langle \text{EoS} \rangle$

tagset $\{\text{A}, \text{B}\}$

| $C_0$ | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $\prod_{i=1}^{l} \lambda_{c_i}^{(c_{i-1})} \times \theta_{w_i}^{(c_i)}$ |
|---|---|---|---|---|---|
| BoS | A | A | A | EoS | |
| BoS | A | A | B | EoS | |
| BoS | A | B | A | EoS | |
| BoS | A | B | B | EoS | |
| BoS | B | A | A | EoS | |
| BoS | B | A | B | EoS | |
| BoS | B | B | A | EoS | |
| BoS | B | B | B | EoS | |

Brute force: enumerate sequences, score, sum. But, as we know, there are K^L sequences!

# Forward Algorithm

For a **given** text $w_{1:l}$,
$\alpha(i, j)$ is the total probability of all
sequences ending in $(C_i = j, W_i = w_i)$:

$$\alpha(i,j) = \begin{cases} \lambda_j^{(\mathrm{BoS})} \theta_{w_i}^{(j)} & \text{if } i = 1 \\ \sum_{r \in \mathcal{C}} \alpha(i-1, r) \lambda_j^{(r)} \theta_{w_i}^{(j)} & \text{if } i > 1 \end{cases}$$

Then, $\alpha(l+1, \mathrm{EOS})$ is the marginal
probability.

Watch the video I prepared for you.

# Value Recursion

Let $s(r, j, w_i)$ be the score associated with setting $C_i = j$ for a given text $X = w_{1:l}$ when $C_{i-1} = r$.

$$\alpha(i, j) = \begin{cases} s(\text{BoS}, j, w_i) & \text{if } i = 1 \\ \bigoplus_{r \in \mathcal{C}} \alpha(i-1, r) \otimes s(r, j, w_i) & \text{if } i > 1 \end{cases}$$

The generalised sum $a \oplus b$ operationalises the semantics of **disjunctions** (i.e., $a$ or $b$).
The generalised product $a \otimes b$ operationalises the semantic s of **conjunctions** (i.e., $a$ and $b$).

Watch the video I prepared for you.

# Value Recursion > Forward

- Forward:
  - $s(r, j, w_i) = \lambda_j^{(r)} \times \theta_{w_i}^{(j)}$
  - $a \oplus b = a + b$
  - $a \otimes b = a \times b$
- Forward (log):
  - $s(r, j, w_i) = \log \lambda_j^{(r)} + \log \theta_{w_i}^{(j)}$
  - $a \oplus b = \text{logsumexp}(a, b) = \log(\exp(a) + \exp(b))$
  - $a \otimes b = a + b$

# Value Recursion > Viterbi

- Viterbi:
  - $s(r, j, w_i) = \lambda_j^{(r)} \times \theta_{w_i}^{(c_i)}$
  - $a \oplus b = \max(a, b)$
  - $a \otimes b = a \times b$
- Viterbi (log):
  - $s(c_{i-1}, c_i, w_i) = \log \lambda_{c_i}^{(c_{i-1})} + \log \theta_{w_i}^{(c_i)}$
  - $a \oplus b = \max(a, b)$
  - $a \otimes b = a + b$

# Outline

- Word classes
- Hidden Markov model
- Evaluation
- **Sequence Labelling**
- Conditional random fields

# Sequence Labelling Tasks

There are various sequence labelling tasks, they are useful on their own and they often have nothing to do with generating text.

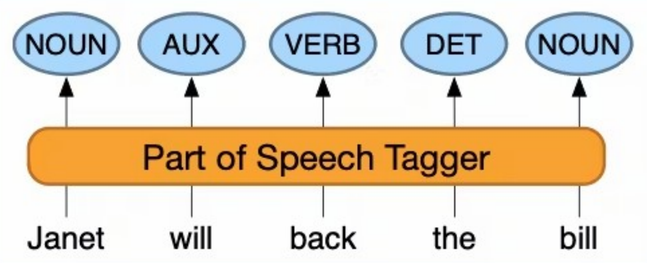



IMAGE FROM JURAFSKY AND MARTIN CH 8

# POS Tagging

We are *given* the text and we do not care to assign probability to it.

Our goal is to develop a system that can POS tag the input sequence.

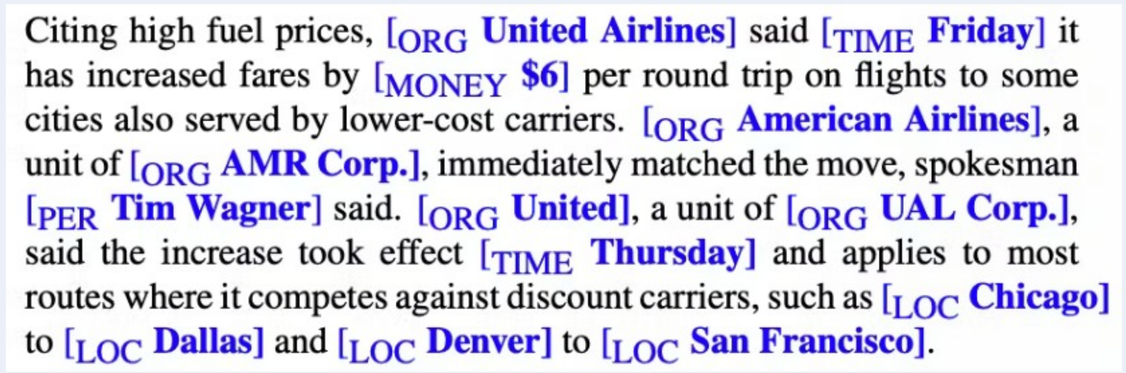



FIGURE FROM JURAFSKY AND MARTIN CH 8

# Named-Entity Recognition

NER is a labelling task from a semantic perspective, where we recognise proper nouns that refer to a certain type of entity.

The text (in black) is *given* and we do not care to assign probability to it. Our goal is to develop a system that can detect and categorise mentions to named entities (i.e., the blue spans)

| Input | Output |
|---|---|
| Jane | B-PER |
| Villanueva | E-PER |
| of | O |
| United | B-ORG |
| Airlines | I-ORG |
| Holding | E-ORG |
| discussed | O |
| the | O |
| Chicago | S-LOC |
| route | O |
| . | O |

# Chunking as Labelling

We can see NER as sequence labelling by labelling tokens as inside or outside a span of text that refers to a named-entity.

(Other annotation schemes are possible, see Section 8.3 of textbook)

> Understanding how participants relate to events—being able to answer the question "Who did what to whom" (and perhaps also "when and where")—is a central question of natural language processing.

– *Jurafsky & Martin, Chapter 24*

- XYZ corporation bought the stock.
- They sold the stock to XYZ corporation.
- The stock was bought by XYZ corporation.
- The purchase of the stock by XYZ corporation...
- The stock purchase by XYZ corporation...

EXAMPLES FROM JURAFSKY & MARTIN CHAPTER 24

# List commonalities between these sentences...

Waiting for responses ...

- XYZ corporation bought the stock.
- They sold the stock to XYZ corporation.
- The stock was bought by XYZ corporation.
- The purchase of the stock by XYZ corporation...
- The stock purchase by XYZ corporation...

EXAMPLES FROM JURAFSKY & MARTIN CHAPTER 24

# Shallow Semantics

→ there was a *purchase* event

→ the participants were *XYZ Corp* and some *stock*

→ *XYZ Corp* was the buyer

→ *stock* was the thing purchased

| Thematic Role | Example |
|---|---|
| AGENT | *The waiter* spilled the soup. |
| EXPERIENCER | *John* has a headache. |
| FORCE | *The wind* blows debris from the mall into our yards. |
| THEME | Only after Benjamin Franklin broke *the ice*... |
| RESULT | The city built a *regulation-size baseball diamond*... |
| CONTENT | Mona asked *"You met Mary Ann at a supermarket?"* |
| INSTRUMENT | He poached catfish, stunning them *with a shocking device*... |
| BENEFICIARY | Whenever Ann Callahan makes hotel reservations *for her boss*... |
| SOURCE | I flew in *from Boston*. |
| GOAL | I drove *to Portland*. |

**Figure 24.2** Some prototypical examples of various thematic roles.

## Prototypical Semantic Roles

EXAMPLE FROM JURAFSKY & MARTIN, CHAPTER 24

# Sequence Role Labelling

(24.3) *John broke the window.*
AGENT THEME

(24.4) *John broke the window with a rock.*
AGENT THEME INSTRUMENT

(24.5) *The rock broke the window.*
INSTRUMENT THEME

(24.6) *The window broke.*
THEME

(24.7) *The window was broken by John.*
THEME AGENT

Assigning semantic roles to spans in sentences.

These semantic roles express the role that arguments of a predicate take in the event.

(24.11) **agree.01**

Arg0: Agreer
Arg1: Proposition
Arg2: Other entity agreeing

Ex1: [$_{\text{Arg0}}$ The group] *agreed* [$_{\text{Arg1}}$ it wouldn't make an offer].
Ex2: [$_{\text{ArgM-TMP}}$ Usually] [$_{\text{Arg0}}$ John] *agrees* [$_{\text{Arg2}}$ with Mary] [$_{\text{Arg1}}$ on everything].

(24.12) **fall.01**

Arg1: Logical subject, patient, thing falling
Arg2: Extent, amount fallen
Arg3: start point
Arg4: end point, end state of arg1
Ex1: [$_{\text{Arg1}}$ Sales] *fell* [$_{\text{Arg4}}$ to $25 million] [$_{\text{Arg3}}$ from $27 million].
Ex2: [$_{\text{Arg1}}$ The average junk bond] *fell* [$_{\text{Arg2}}$ by 4.2%].

FROM JURAFSKY & MARTIN, CHAPTER 4

# SRL Examples

The semantics of the roles depend on the verb and its sense as codified in databases like PropBank and FrameNet.

| word | POS | frame file | roleset | #1 (have.01) | #2 (like.02) | #3 (lighten_up.02) | #4 (expose.01) | #5 (use.01) | #6 (call.03) | #7 implement.01) |
|---|---|---|---|---|---|---|---|---|---|---|
| I | PRP | - | - | * | * | * | * | (ARG0*) | * | (ARG0*) |
| have | VBP | have | have.01 | (V*) | * | * | * | * | * | * |
| price | NN | - | - | * | * | * | * | * | * | * |
| targets | NNS | - | - | * | * | * | * | * | * | * |
| of | IN | - | - | * | * | * | * | * | * | * |
| where | WRB | - | - | * | (ARGM-LOC*) | * | * | * | * | * |
| I | PRP | - | - | * | (ARG0*) | (ARG0*) | * | * | * | * |
| would | MD | - | - | * | (ARGM-MOD*) | * | * | * | * | * |
| like | VB | liken | like.02 | * | (V*) | * | * | * | * | * |
| to | TO | - | - | * | (ARG1* | * | * | * | * | * |
| lighten | VB | lighten | lighten_up.02 | * | * | (V* | * | * | * | * |
| up | RP | - | - | * | * | *) | * | * | * | * |
| exposure | NN | expose | expose.01 | * | * | (ARG1* | (V*) | * | * | * |
| to | IN | - | - | * | * | * | (ARG2* | * | * | * |
| ENE | NNP | - | - | * | *) | *) | *) | * | * | * |
| and | CC | - | - | * | * | * | * | * | * | * |
| will | MD | - | - | * | * | * | * | (ARGM-MOD*) | * | * |
| use | VB | use | use.01 | * | * | * | * | (V*) | * | * |
| calls | NNS | call | call.03 | * | * | * | * | (ARG1*) | (V*) | * |
| to | TO | - | - | * | * | * | * | (ARG2* | * | * |
| implement | VB | implement | implement.01 | * | * | * | * | * | * | (V*) |
| the | DT | - | - | * | * | * | * | * | * | (ARG1* |
| stategy | NN | - | - | * | * | * | * | *) | * | *) |
| . | . | - | - | * | * | * | * | * | * | * |

Semantic roles as sequence labelling: each column (after *roleset*) is the target sequence wrt a given predicate. Example from PropBank.

| Input | | Output |
|---|---|---|
| position | word | SRL (t=18) |
| 1 | I | S-A0 |
| 2 | have | O |
| 3 | price | O |
| 4 | targets | O |
| 5 | of | O |
| 6 | where | O |
| 7 | I | O |
| 8 | would | O |
| 9 | like | O |
| 10 | to | O |
| 11 | lighten | O |
| 12 | up | O |
| 13 | exposure | O |
| 14 | to | O |
| 15 | ENE | O |
| 16 | and | O |
| 17 | will | S-AMOD |
| 18 | use | S-V |
| 19 | calls | S-A1 |
| 20 | to | B-A2 |
| 21 | implement | I-A2 |
| 22 | the | I-A2 |
| 23 | stategy | E-A2 |
| 24 | . | O |

# SRL using IOBES

For each predicate (the example has 7), we create an input-output pair.

The input is a sequence of words $w_{1:l}$ and the position $t$ of the verb predicate whose semantic arguments we analyse.

The output $c_{1:l}$ is the IOBES-encoded sequence of semantic arguments of the verb predicate $w_t$.

# HMM for Sequence Labelling

If we can express the task as annotating the tokens in a sequence of size $L$, each with a category (out of a finite set), then the HMM is readily applicable.

Good examples: POS tagging, NER.
Not so good example: in SRL, the number of output tag-sequences depends on the number of predicates in the input.

But is the HMM a good choice for those good examples?

# Limitations of the HMM (in particular, given that our application does not need to assign probability to text)?

Waiting for responses ...

# Key Technical Limitation

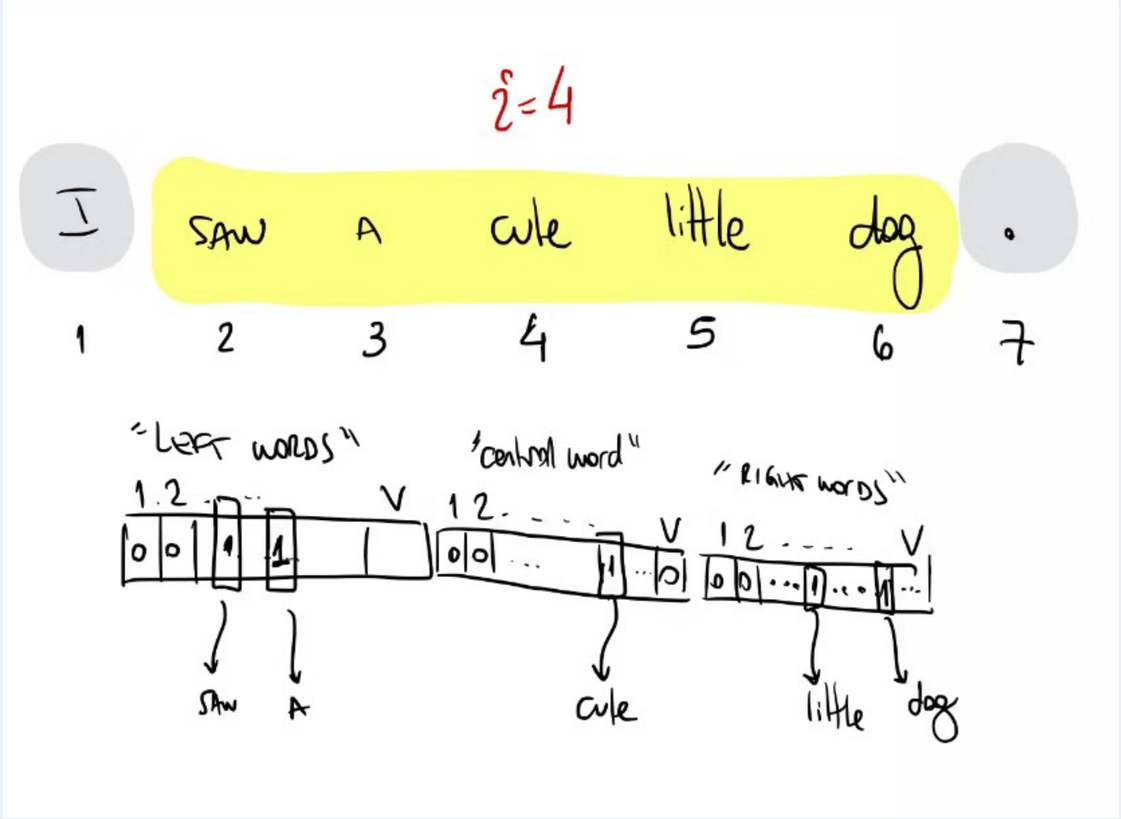
Because HMMs need to generate text, they power sequence labellers that make fairly limited use of linguistic context in $w_{1:l}$.

Having $C_i$ interact with words other than $W_i$ would make key quantities in the HMM *very* hard to compute (e.g., marginal and mode probabilities). It would also make the tabular CPDs rather sparse.

# Limitations from a Linguistic Perspective

Unseen words and phrases (e.g., proper names and acronyms, inflected verbs, phrasal verbs) are actually quite frequent.

In many cases, their likely interpretation (e.g., syntactic or semantic function) are identifiable from fine-grained features: capitalisation (in English), prefixes and suffixes (e.g., "un-" or "-ed"), knowing the words surrounding a certain position (e.g., a window of 5 words), etc.

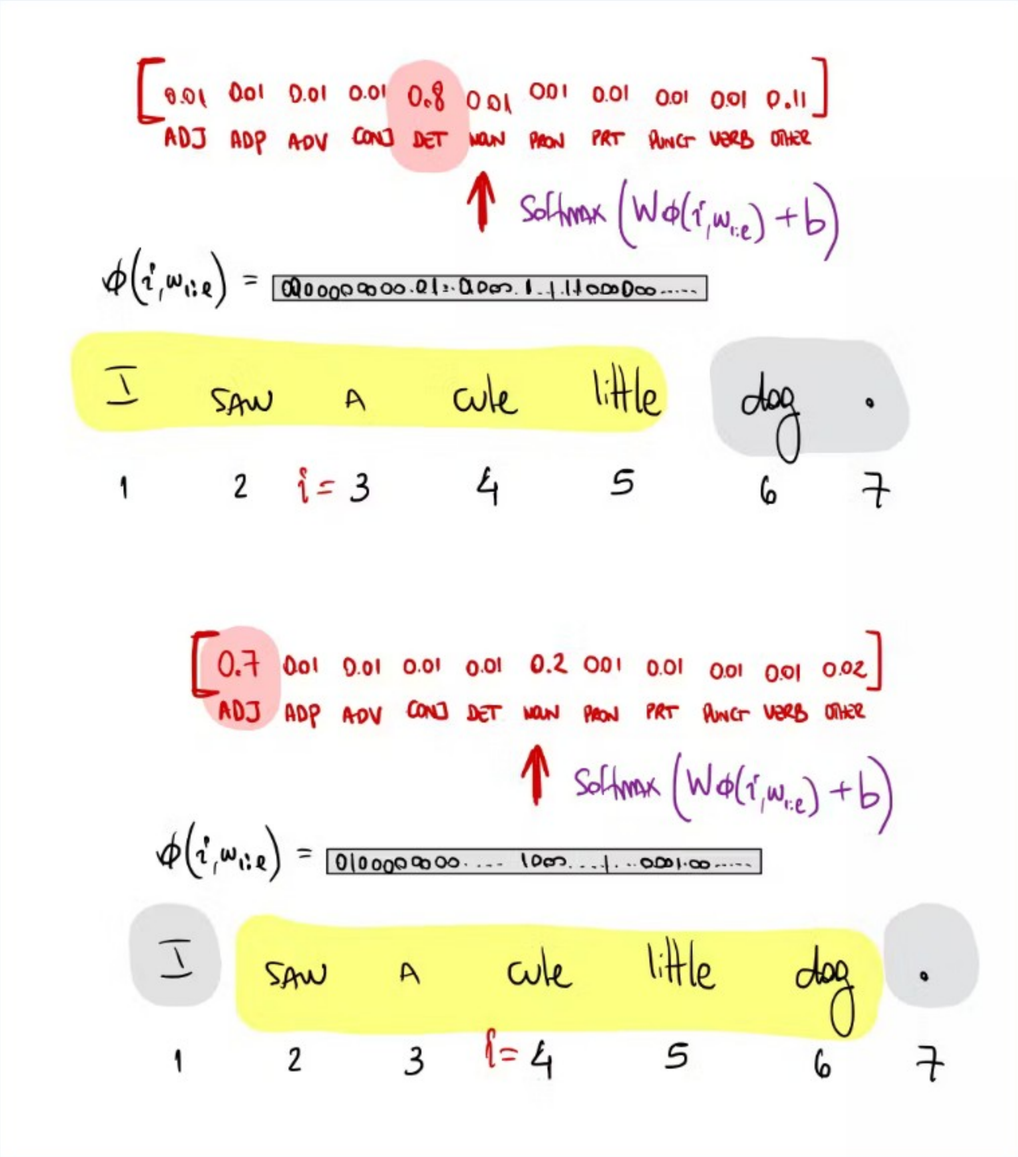



# First Idea: use feature-rich models

Let's introduce a feature function $\phi(i, w_{1:l})$ to *represent* the context in which we predict the distribution of the $i$th tag in the output tag-sequence.

The feature function is a design choice, it should express what is known about the $i$h position of the sequence.

Example: $V$ BoW features for the left-neighbourhood of the $i$th position, same for the right-neighbourhood, and a $V$-dimensional indicator for the central word (position $i$).




# Second Idea: design one classifier and use it many times

Given the text $w_{1:l}$, we represent the $i$th position using a $D$-dimensional feature vector $\boldsymbol{\phi}(i, w_{1:l})$ and predict the Categorical distribution of the $i$th tag in the tag-sequence with a log-linear model:

$$\mathbf{s} = \mathbf{W}\,\boldsymbol{\phi}(i, w_{1:l}) + \mathbf{b}$$
$$\mathbf{f}(w_{1:l}, i; \boldsymbol{\theta}) = \mathrm{softmax}(\mathbf{s})$$

with $\boldsymbol{\theta} = \{\mathbf{b} \in \mathbb{R}^K, \mathbf{W} \in \mathbb{R}^{K \times D}\}$

# Independent Tagger

Model positions in the tag-sequence independently of one another. This is basically treating tagging as a *chain* of independent applications of the same text classifier.

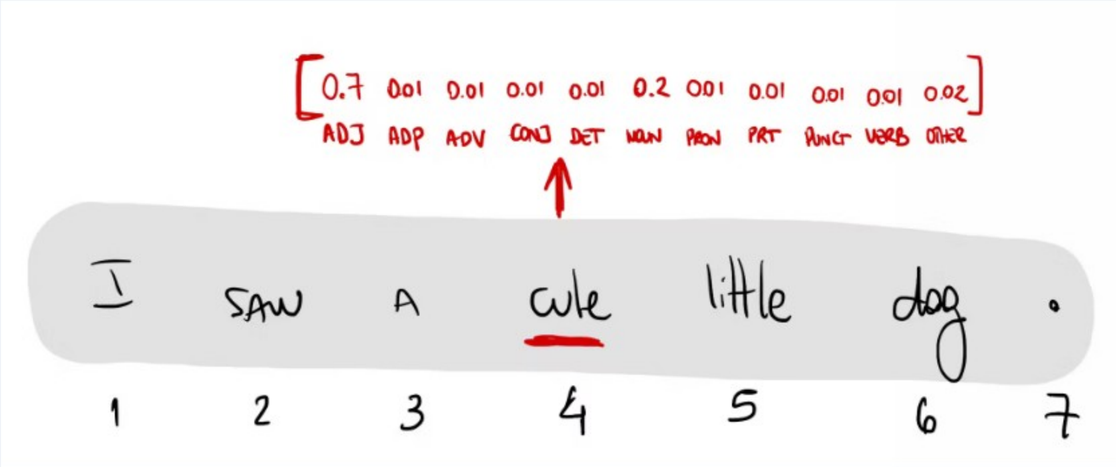
$$C_i | X = w_{1:l}, I = i \sim \text{Categorical}(\mathbf{f}(w_{1:l}, i; \boldsymbol{\theta}))$$

For parameter estimation, watch the video on logistic CPDs.

# Parameter estimation

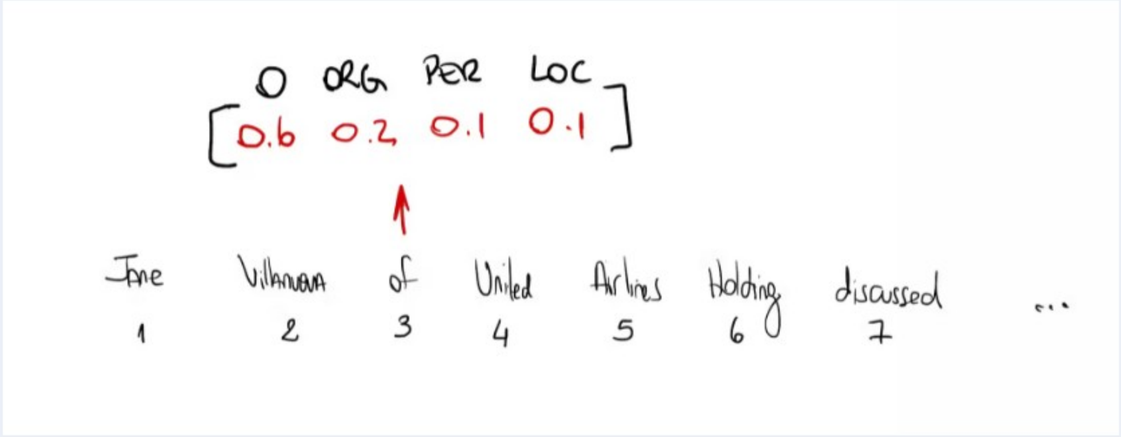
The log-linear model assigns probability $f_k(w_{1:l}, i; \boldsymbol{\theta})$ to classifying $w_i$ in $w_{1:l}$ as $k$.
We can obtain a parameter estimate by following the direction of steepest ascent using the gradient:
$\nabla_\theta \log f_k(w_{1:l}, i; \boldsymbol{\theta})$.

The different steps of a sequence are treated as if they were independent training examples.




# Example: POS classifier

Given an input text (e.g., *I saw a cute little dog.*) and a position (e.g., *4*) we want to analyse using POS, we predict a distribution over the tagset (e.g., {ADJ, ADP, ADV, CONJ, DET, NOUN, PRON, PRT, PUNCT, VERB, OTHER}).




# Example: NE classifier

Given an input text (e.g., *Jane Viallnueva of United Airlines Holding discussed...*) and a position (e.g., *4*) we want to analyse in terms of NE, we predict a distribution over the tagset (e.g., {O, ORG, PER, LOC}).

| Input | Output |
|---|---|
| Jane | B-PER |
| Villanueva | E-PER |
| of | O |
| United | B-ORG |
| Airlines | I-ORG |
| Holding | E-ORG |
| discussed | O |
| the | O |
| Chicago | S-LOC |
| route | O |
| . | O |

IOBES REPRESENTATION FOR NER DATA

# What could go wrong with independent tagging?

Waiting for responses ...

# Questions about sequence labelling tasks?

4 questions
7 upvotes

# Outline

- Word classes
- Hidden Markov model
- Evaluation
- Sequence Labelling
- **Conditional random fields**


# Another way to prescribe a feature-rich model

Suppose we assign scores to complete tag-sequences. Then we have the conditional probability $P_{Y|X}(c_{1:l}|w_{1:l})$ be proportional to $\exp(\text{score}(c_{1:l}, w_{1:l}))$.

The exp makes this function $\geq 0$, with $0$ only possible when the score is $-\infty$ (reserved for invalid sequences).

# How can we get a valid probability distribution for $P_{Y|X}(c_{1:l}|w_{1:l}) \propto \exp(\text{score}(c_{1:l}, w_{1:l}))$

Waiting for responses ...


# Another way to prescribe a feature-rich model

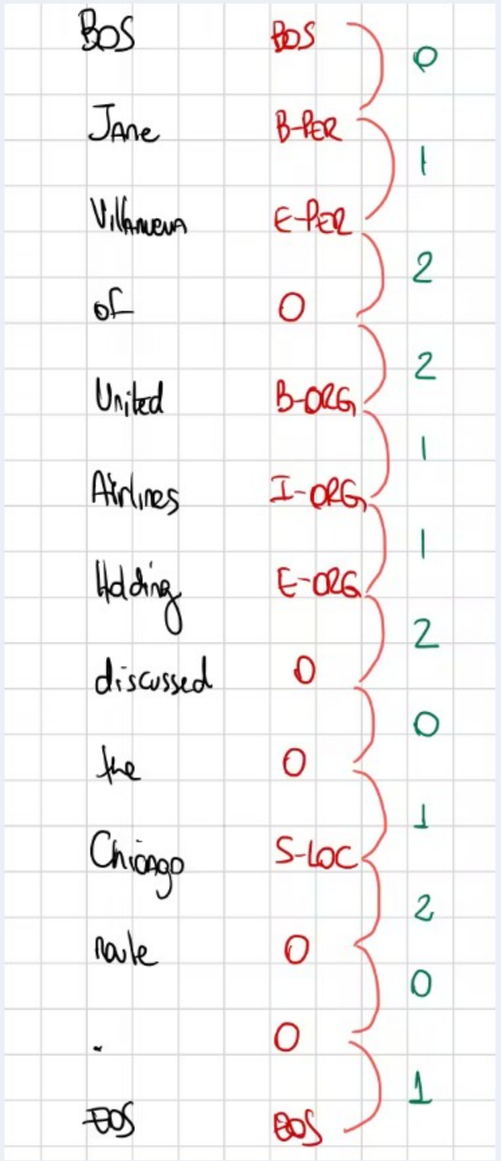
Let $\mathcal{Z}(w_{1:l}) = \sum_{o_{1:l} \in \mathcal{C}^l} \exp(\text{score}(o_{1:l}, w_{1:l}))$ be the sum of exponentiated scores for all possible sequences.

Then, $P_{Y|X}(c_{1:l}|w_{1:l}) = \dfrac{\exp(\text{score}(c_{1:l}, w_{1:l}))}{\mathcal{Z}(w_{1:l})}$.

This works provided we *can* compute the normaliser.

# What could make $\sum_{o_{1:l} \in \mathcal{C}^l} \exp(\text{score}(o_{1:l}, w_{1:l}))$ tractable to compute?

Waiting for responses ...

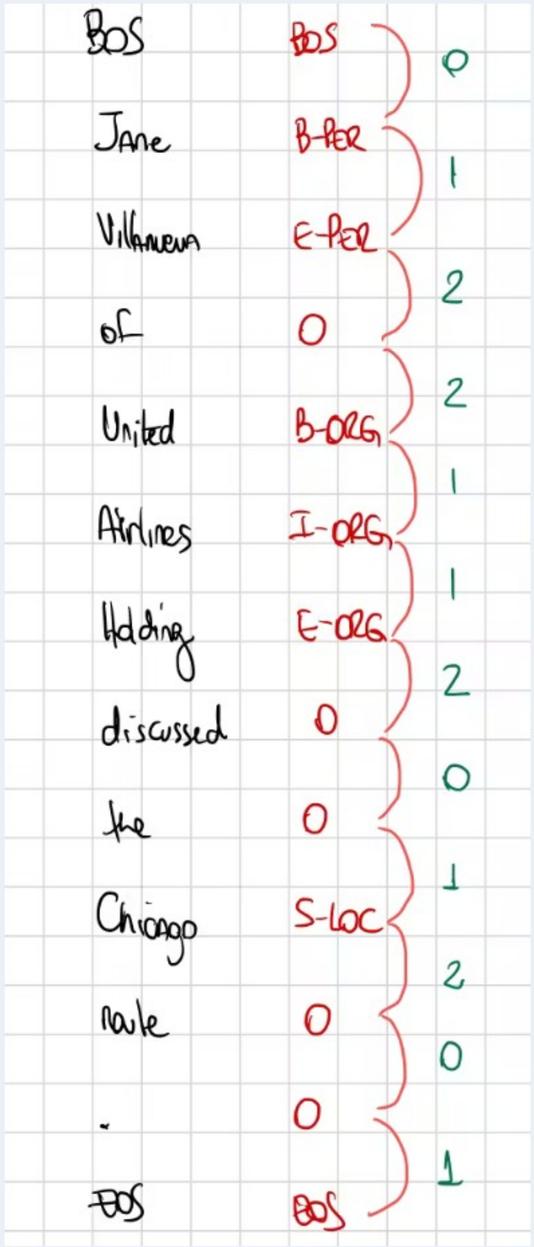

# Score Decomposition

Step: tag the $i$th token in $w_{1:l}$ with $c$ when the previous token was tagged with $r$.

Assign scores to steps and have the score of a complete sequence be

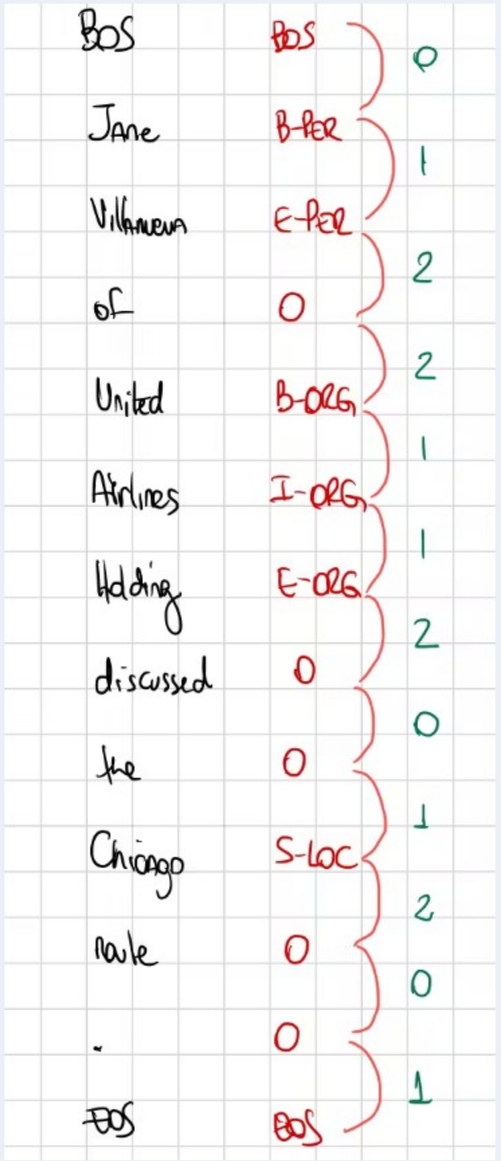
$$\text{score}(c_{1:l}, w_{1:l}) = \sum_{i=1}^{l} \text{scorestep}(r, c, i, w_{1:l})$$

This corresponds to a form of conditional independence assumption in a undirected graphical model.




# Where do we get scores from?

Waiting for responses ...




## We learn to score steps:

- $\text{scorestep}(r, c, i, w_{1:l}) = \mathbf{w}^\top \boldsymbol{\phi}(c_{i-1}, c_i, i, w_{1:l}) + b$
- $r$ is the tag at position $i - 1$
- $c$ is the tag at position $i$
- $\boldsymbol{\phi}(r, c, i, w_{1:l})$ is a $D$-dimensional feature representation for the step
- $\boldsymbol{\theta} = \{\mathbf{w} \in \mathbb{R}^D, b \in \mathbb{R}\}$ are trainable parameters

# Linear-Chain Conditional Random Field

The pmf of the linear-chain CRF assigns probability:

$$p(c_{1:l}|w_{1:l};\boldsymbol{\theta}) = \frac{\sum_{i=1}^{l} \mathbf{w}^\top \phi(c_{i-1}, c_i, i, w_{1:l}) + b}{\mathcal{Z}(w_{1:l};\boldsymbol{\theta})}$$

to a tag-sequence $c_{1:l}$ given a token-sequence $w_{1:l}$. The denominator

$$\mathcal{Z}(w_{1:l};\boldsymbol{\theta}) = \sum_{o_{1:l} \in \mathcal{C}^l} \exp\left(\sum_{i=1}^{l} \mathbf{w}^\top \phi(o_{i-1}, o_i, i, w_{1:l}) + b\right)$$

can be computed using the value recursion.

# Parameter Estimation

Same as any other log-linear model: gradient-based optimisation.

For the likelihood function we need to be able to assess the probability mass of observed $c_{1:l}$ given observed $w_{1:l}$ as a function of model parameters:

$$\log p(c_{1:l}|w_{1:l};\boldsymbol{\theta}) = \left(\sum_{i=1}^{l} \mathbf{w}^\top \phi(c_{i-1}, c_i, i, w_{1:l}) + b\right) - \alpha(\mathrm{EOS}, l+1)$$

For $\alpha(\cdot)$ the value recursion using
$s(r, c, i, w_{1:l}) = \mathbf{w}^\top \phi(r, c, i, w_{1:l}) + b$,
$\oplus = \mathrm{logsumexp}$ and $\otimes = +$

| Input position | word | Output |
|---|---|---|
| 1 | I | S-A0 |
| 2 | have | O |
| 3 | price | O |
| 4 | targets | O |
| 5 | of | O |
| 6 | where | O |
| 7 | I | O |
| 8 | would | O |
| 9 | like | O |
| 10 | to | O |
| 11 | lighten | O |
| 12 | up | O |
| 13 | exposure | O |
| 14 | to | O |
| 15 | ENE | O |
| 16 | and | O |
| 17 | will | S-AMOD |
| 18 | use | S-V |
| 19 | calls | S-A1 |
| 20 | to | B-A2 |
| 21 | implement | I-A2 |
| 22 | the | I-A2 |
| 23 | stategy | E-A2 |
| 24 | . | O |

| Input position | word | Output |
|---|---|---|
| 1 | I | S-A0 |
| 2 | have | O |
| 3 | price | O |
| 4 | targets | O |
| 5 | of | O |
| 6 | where | O |
| 7 | I | O |
| 8 | would | O |
| 9 | like | O |
| 10 | to | O |
| 11 | lighten | O |
| 12 | up | O |
| 13 | exposure | O |
| 14 | to | O |
| 15 | ENE | O |
| 16 | and | O |
| 17 | will | O |
| 18 | use | O |
| 19 | calls | O |
| 20 | to | O |
| 21 | implement | S-V |
| 22 | the | B-A1 |
| 23 | stategy | E-A1 |
| 24 | . | O |

# How about Multiple Output Sequences?

Recall that in SRL we have multiple output sequences?

Extend the CRF with one additional input $t$ indicating which predicate we are tagging. Then use the same CRF for each of the predicates.

$$P_{Y|XT}(c_{1:l}|w_{1:l}, t) \propto \exp\left(\sum_{i=1}^{l} \mathbf{w}^\top \phi(c_{i-1}, c_i, i, t, w_{1:l}) + b\right)$$

# Limitations and Improvements

- Designing good feature functions can be difficult
  - See textbook for examples
- Interesting feature spaces are often **huge** and super **sparse**
  - It's often easier and more technically feasible to *learn* compact and expressive features (with NNs)
  - e.g., linear-chain CRFs whose local scores are predicted by powerful NNs
- Structural constraints must be expressed locally (by adjacent tags)
  - Some advanced (e.g., autoregressive and energy-based) models can circumvent this

# Questions about the CRF?

Waiting for responses ...

# Summary

- We can generalise words into classes, alleviating data sparsity
- HMM generates sequence pairs with strong Markov assumptions
- Value recursion enables efficient inference for HMMs
- HMMs can power POS, NER, but suffer from limited use of linguistic context
- Sequence labellers can be designed without generating text
- Feature-based models enable better use of linguistic context
- Tagging can be seen as a chain of simple classification steps
- CRFs allow for structure prediction with more realistic Markov assumptions

# What next?

- Required: watch the value recursion video
  - ask me questions in class or on Piazza
- Next class: syntactic parsing
  - please check background material