Part Of Speech Tagging

Part-of-Speech Tagging

INPUT:
Profits soared at Boeing Co., easily topping forecasts on Wall Street, as their CEO Alan Mulally announced first quarter results.

OUTPUT:
Profits/N soared/V at/P Boeing/N Co./N ,/, easily/ADV topping/V forecasts/N on/P Wall/N Street/N ,/, as/P their/POSS CEO/N Alan/N Mulally/N announced/V first/ADJ quarter/N results/N ./.

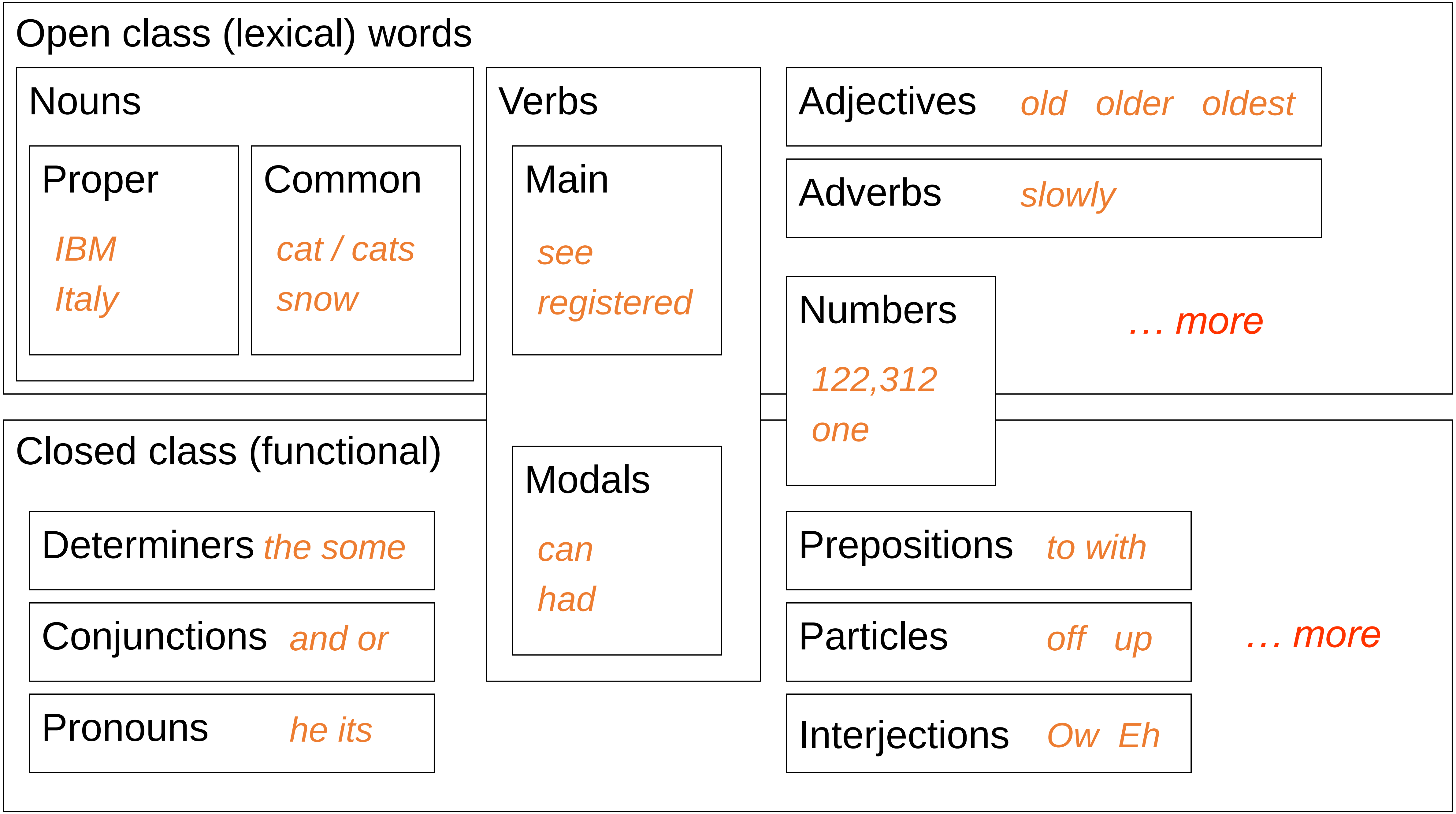
N = Noun
V = Verb
P = Preposition
Adv = Adverb
Adj = Adjective
...

Our Goal

Training set:

1 Pierre/NNP Vinken/NNP ,/, 61/CD years/NNS old/JJ ,/, will/MD join/VB the/DT board/NN as/IN a/DT nonexecutive/JJ director/NN Nov./NNP 29/CD ./.
2 Mr./NNP Vinken/NNP is/VBZ chairman/NN of/IN Elsevier/NNP N.V./NNP ,/, the/DT Dutch/NNP publishing/VBG group/NN ./.
3 Rudolph/NNP Agnew/NNP ,/, 55/CD years/NNS old/JJ and/CC chairman/NN of/IN Consolidated/NNP Gold/NNP Fields/NNP PLC/NNP ,/, was/VBD named/VBN a/DT nonexecutive/JJ director/NN of/IN this/DT British/JJ industrial/JJ conglomerate/NN ./.
...
38,219 It/PRP is/VBZ also/RB pulling/VBG 20/CD people/NNS out/IN of/IN Puerto/NNP Rico/NNP ,/, who/WP were/VBD helping/VBG Huricane/NNP Hugo/NNP victims/NNS ,/, and/CC sending/VBG them/PRP to/TO San/NNP Francisco/NNP instead/RB ./.

- From the training set, induce a function/algorithm that maps new sentences to their tag sequences.


Open class (lexical) words
Nouns
Proper
IBM
Italy
Common
cat / cats
snow
Verbs
Main
see
registered
Modals
can
had
Adjectives old older oldest
Adverbs slowly
Numbers
122,312
one
… more
Closed class (functional)
Determiners the some
Conjunctions and or
Pronouns he its
Prepositions to with
Particles off up
Interjections Ow Eh
… more

Open vs. Closed classes

- Open vs. Closed classes
  - Closed:
    - determiners: ***a, an, the***
    - pronouns: ***she, he, I***
    - prepositions: ***on, under, over, near, by, …***
    - Why “closed”?
  - Open:
    - Nouns, Verbs, Adjectives, Adverbs.

POS Tagging

- Words often have more than one POS: *back*
  - The *back* door = JJ
  - On my *back* = NN
  - Win the voters *back* = RB
  - Promised to *back* the bill = VB
- The POS tagging problem is to determine the POS tag for a particular instance of a word.

Two Types of Constraints

Influential/JJ members/NNS of/IN the/DT House/NNP Ways/NNP and/CC Means/NNP Committee/NNP introduced/VBD legislation/NN that/WDT would/MD restrict/VB how/WRB the/DT new/JJ savings-and-loan/NN bailout/NN agency/NN can/MD raise/VB capital/NN ./.

- "Local": e.g., *can* is more likely to be a modal verb MD rather than a noun NN
- "Contextual": e.g., a noun is much more likely than a verb to follow a determiner
- Sometimes these preferences are in conflict:

  The trash can is in the garage

POS tagging performance

- How many tags are correct? (Tag accuracy)
  - About 97% currently
  - But baseline is already 90%
    - Baseline is performance of stupidest possible method
      - Tag every word with its most frequent tag
      - Tag unknown words as nouns
  - Partly easy because
    - Many words are unambiguous
    - You get points for them (*the*, *a*, etc.) and for punctuation marks!

How difficult is POS tagging?

- About 11% of the word types in the Brown corpus are ambiguous with regard to part of speech
- But they tend to be very common words. E.g., *that*
  - I know *that* he is honest = IN
  - Yes, *that* play was nice = DT
  - You can’t go *that* far = RB
- 40% of the word tokens are ambiguous

Sources of information

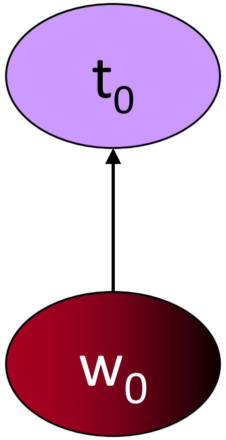
- What are the main sources of information for POS tagging?
  - Knowledge of neighboring words
    - Bill saw that man yesterday
    - NNP NN DT NN NN
    - VB VB(D) IN VB NN
  - Knowledge of word probabilities
    - *man* is rarely used as a verb….
- The latter proves the most useful, but the former also helps

More and Better Features ➔ Feature-based tagger

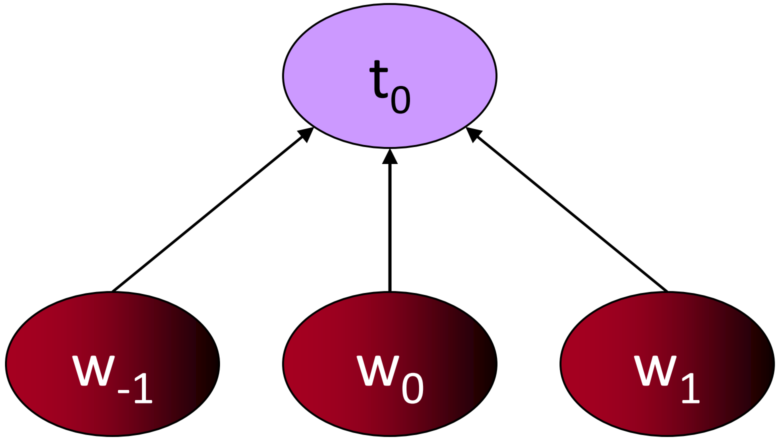
- Can do surprisingly well just looking at a word by itself:
  - Word   the: the $\rightarrow$ DT
  - Lowercased word   Importantly: importantly $\rightarrow$ RB
  - Prefixes   unfathomable: un- $\rightarrow$ JJ
  - Suffixes   Importantly: -ly $\rightarrow$ RB
  - Capitalization   Meridian: CAP $\rightarrow$ NNP
  - Word shapes   35-year: d-x $\rightarrow$ JJ
- Then build a supervised machine learning model to predict tag
  - P(t|w):   93.7% overall

Tagging Without Sequence Information

Baseline



Three Words



Model	Features	Token	Unknown	Sentence
Baseline	56,805	**93.69%**	82.61%	26.74%
3Words	239,767	**96.57%**	86.78%	48.27%

Using words only in a straight classifier works as well as a basic (HMM) sequence model!!

HMM Part of Speech Tagging

- The HMM is a sequence model.
- A sequence model or sequence classifier is a model whose job is to assign a label or class to each unit in a sequence, thus mapping a sequence of observations to a sequence of labels.

Markov Chains

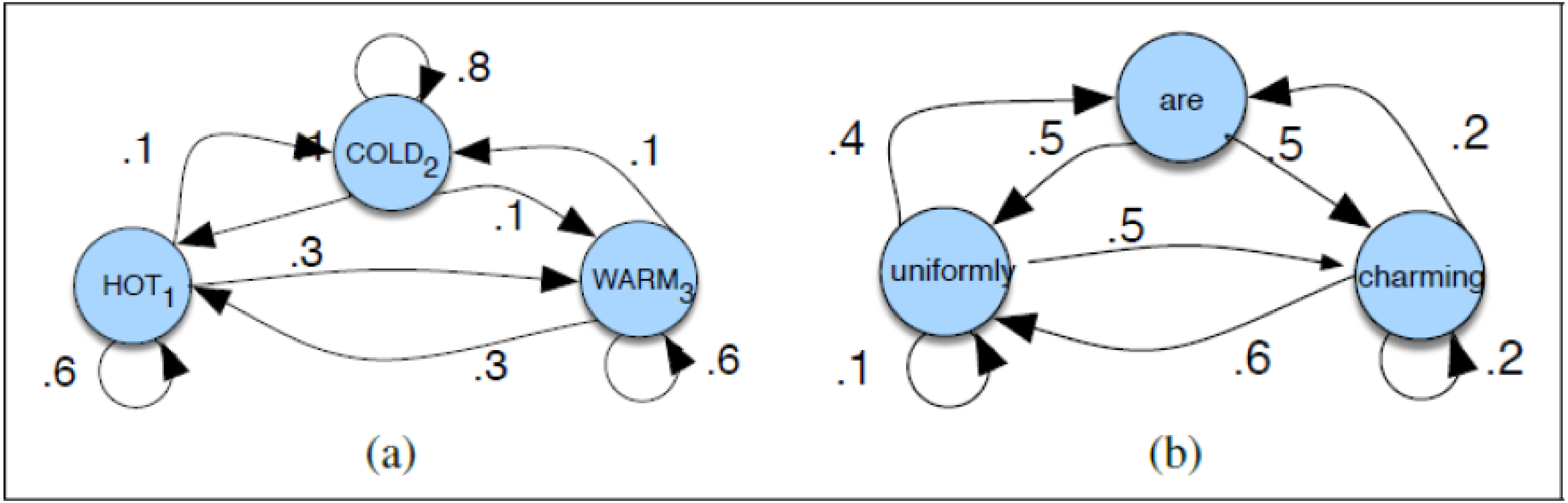


Figure 8.3 A Markov chain for weather (a) and one for words (b), showing states and transitions. A start distribution π is required; setting $\pi = [0.1, 0.7, 0.2]$ for (a) would mean a probability 0.7 of starting in state 2 (cold), probability 0.1 of starting in state 1 (hot), etc.

Components of Markov Chain

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} a_{12} \ldots a_{n1} \ldots a_{nn}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$\pi = \pi_1, \pi_2, \ldots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

Hidden Markov Model

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} \ldots a_{ij} \ldots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \ldots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, ..., v_V$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state i
$\pi = \pi_1, \pi_2, ..., \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

First Order Hidden Markov Model

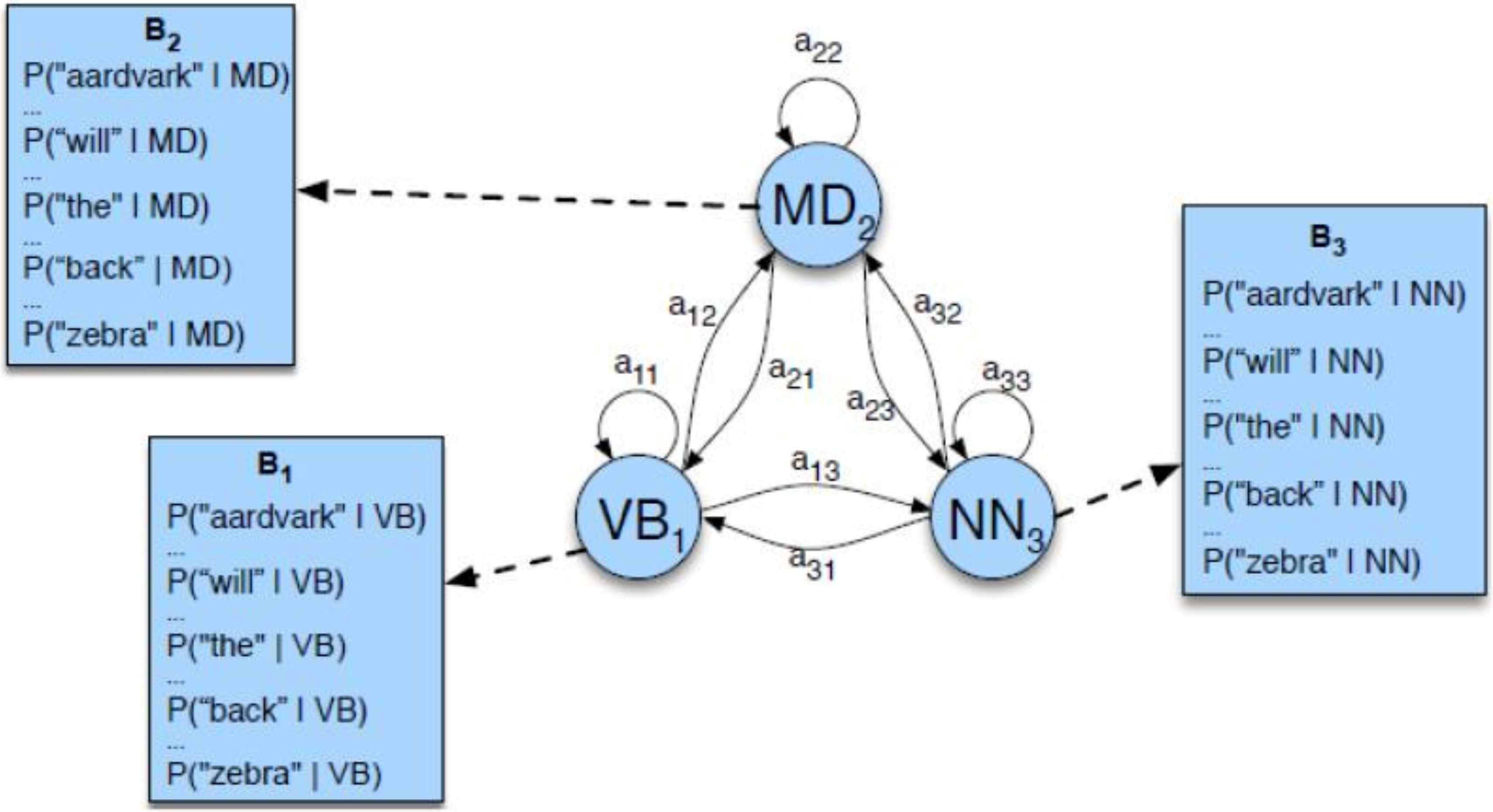


Figure 8.4 An illustration of the two parts of an HMM representation: the *A* transition probabilities used to compute the prior probability, and the *B* observation likelihoods that are associated with each state, one likelihood for each possible observation word.

Hidden Markov Models

- We have an input sentence $x = x_1, x_2, \ldots, x_n$
  (x_i is the i'th word in the sentence)

- We have a tag sequence $y = y_1, y_2, \ldots, y_n$
  (y_i is the i'th tag in the sentence)

- We'll use an HMM to define

$$p(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n)$$

  for any sentence $x_1 \ldots x_n$ and tag sequence $y_1 \ldots y_n$ of the same length.

- Then the most likely tag sequence for x is

$$\arg\max_{y_1 \ldots y_n} p(x_1 \ldots x_n, y_1, y_2, \ldots, y_n)$$

Hidden Markov Model

Zero Order Markov Model (Unigram Model)

$$p(x_1 \ldots x_n y_i \ldots y_{n+1}) = \prod_{i=1}^{n+1} q(y_i) \prod_{i=1}^{n} e(x_i | \, y_i)$$

First Order Markov Model (Bigram Model)

$$p(x_1 \ldots x_n y_i \ldots y_{n+1}) = \prod_{i=1}^{n+1} q(y_i | \, y_{i-1}) \prod_{i=1}^{n} e(x_i | \, y_i)$$

Trigram Hidden Markov Models (Trigram HMMs)

For any sentence $x_1 \ldots x_n$ where $x_i \in \mathcal{V}$ for $i = 1 \ldots n$, and any tag sequence $y_1 \ldots y_{n+1}$ where $y_i \in \mathcal{S}$ for $i = 1 \ldots n$, and $y_{n+1} =$ STOP, the joint probability of the sentence and tag sequence is

$$p(x_1 \ldots x_n, y_1 \ldots y_{n+1}) = \prod_{i=1}^{n+1} q(y_i | y_{i-2}, y_{i-1}) \prod_{i=1}^{n} e(x_i | y_i)$$

where we have assumed that $x_0 = x_{-1} =$ *.

Parameters of the model:

- $q(s|u, v)$ for any $s \in \mathcal{S} \cup \{\text{STOP}\}$, $u, v \in \mathcal{S} \cup \{*\}$
- $e(x|s)$ for any $s \in \mathcal{S}$, $x \in \mathcal{V}$

An Example

If we have $n = 3$, $x_1 \ldots x_3$ equal to the sentence *the dog laughs*, and $y_1 \ldots y_4$ equal to the tag sequence D N V STOP, then

$$\begin{aligned} & p(x_1 \ldots x_n, y_1 \ldots y_{n+1}) \\ = \; & q(\mathtt{D}|*,*) \times q(\mathtt{N}|*,\mathtt{D}) \times q(\mathtt{V}|\mathtt{D},\mathtt{N}) \times q(\mathtt{STOP}|\mathtt{N},\mathtt{V}) \\ & \times e(\textit{the}|\mathtt{D}) \times e(\textit{dog}|\mathtt{N}) \times e(\textit{laughs}|\mathtt{V}) \end{aligned}$$

- STOP is a special tag that terminates the sequence
- We take $y_0 = y_{-1} = *$, where * is a special "padding" symbol

Why the Name?

$$p(x_1 \ldots x_n, y_1 \ldots y_n) = \underbrace{q(\mathsf{STOP}|y_{n-1}, y_n) \prod_{j=1}^{n} q(y_j \mid y_{j-2}, y_{j-1})}_{\text{Markov Chain}}$$

$$\times \underbrace{\prod_{j=1}^{n} e(x_j \mid y_j)}_{x_j\text{'s are observed}}$$

Smoothed Estimation

$$\begin{aligned} q(\mathsf{Vt} \mid \mathsf{DT},\ \mathsf{JJ}) &= \lambda_1 \times \frac{\mathsf{Count}(\mathsf{Dt},\ \mathsf{JJ},\ \mathsf{Vt})}{\mathsf{Count}(\mathsf{Dt},\ \mathsf{JJ})} \\ &\quad + \lambda_2 \times \frac{\mathsf{Count}(\mathsf{JJ},\ \mathsf{Vt})}{\mathsf{Count}(\mathsf{JJ})} \\ &\quad + \lambda_3 \times \frac{\mathsf{Count}(\mathsf{Vt})}{\mathsf{Count}()} \end{aligned}$$

$$\lambda_1 + \lambda_2 + \lambda_3 = 1, \quad \text{and for all } i,\ \lambda_i \geq 0$$

$$e(\mathsf{base} \mid \mathsf{Vt}) = \frac{\mathsf{Count}(\mathsf{Vt},\ \mathsf{base})}{\mathsf{Count}(\mathsf{Vt})}$$

Named Entity Recognition

INPUT: Profits soared at Boeing Co., easily topping forecasts on Wall Street, as their CEO Alan Mulally announced first quarter results.

OUTPUT: Profits soared at [Company Boeing Co.], easily topping forecasts on [Location Wall Street], as their CEO [Person Alan Mulally] announced first quarter results.

Named Entity Extraction as Tagging

INPUT:

Profits soared at Boeing Co., easily topping forecasts on Wall Street, as their CEO Alan Mulally announced first quarter results.

OUTPUT:

Profits/NA soared/NA at/NA Boeing/SC Co./CC ,/NA easily/NA topping/NA forecasts/NA on/NA Wall/SL Street/CL ,/NA as/NA their/NA CEO/NA Alan/SP Mulally/CP announced/NA first/NA quarter/NA results/NA ./NA

NA	= No entity
SC	= Start Company
CC	= Continue Company
SL	= Start Location
CL	= Continue Location

...

Dealing with Low-Frequency Words: An Example

Profits soared at Boeing Co. , easily topping forecasts on Wall Street , as their CEO Alan Mulally announced first quarter results .

Dealing with Low-Frequency Words

A common method is as follows:

- **Step 1**: Split vocabulary into two sets

  Frequent words = words occurring ≥ 5 times in training
  Low frequency words = all other words

- **Step 2**: Map low frequency words into a small, finite set, depending on prefixes, suffixes etc.

Dealing with Low-Frequency Words: An Example

[Bikel et. al 1999] **(named-entity recognition)**

Word class	Example	Intuition
twoDigitNum	90	Two digit year
fourDigitNum	1990	Four digit year
containsDigitAndAlpha	A8956-67	Product code
containsDigitAndDash	09-96	Date
containsDigitAndSlash	11/9/89	Date
containsDigitAndComma	23,000.00	Monetary amount
containsDigitAndPeriod	1.00	Monetary amount, percentage
othernum	456789	Other number
allCaps	BBN	Organization
capPeriod	M.	Person name initial
firstWord	first word of sentence	no useful capitalization information
initCap	Sally	Capitalized word
lowercase	can	Uncapitalized word
other	,	Punctuation marks, all other words

Dealing with Low-Frequency Words: An Example

Profits/NA soared/NA at/NA Boeing/SC Co./CC ,/NA easily/NA topping/NA forecasts/NA on/NA Wall/SL Street/CL ,/NA as/NA their/NA CEO/NA Alan/SP Mulally/CP announced/NA first/NA quarter/NA results/NA ./NA

$\Downarrow$

`firstword`/NA soared/NA at/NA `initCap`/SC Co./CC ,/NA easily/NA `lowercase`/NA forecasts/NA on/NA `initCap`/SL Street/CL ,/NA as/NA their/NA CEO/NA Alan/SP `initCap`/CP announced/NA first/NA quarter/NA results/NA ./NA

NA	= No entity
SC	= Start Company
CC	= Continue Company
SL	= Start Location
CL	= Continue Location

...

Unknown Words

- strongest source of information for guessing the part-of-speech of unknown words is morphology.
- Words that end in -s are likely to be plural nouns (NNS),
- words ending with -ed tend to be past participles (VBN),
- words ending with -able adjectives (JJ),

Unknown Words

- Store for each final letter sequence (word suffixes) of up to 10 letters, the statistics of the tag it was associated with in training.
- We are thus computing for each suffix of length i the probability of the tag t_i given the suffix letters

$$P(t_i|l_{n-i+1} \ldots l_n)$$

- Back-off is used to smooth these probabilities with successively shorter suffixes.

Unknown Words

- Because unknown words are unlikely to be closed-class words like prepositions, suffix probabilities can be computed only for words whose training set frequency is 10, or only for open-class words.

Unknown Words

- we can compute the likelihood $p(w_i|t_i)$ (Prob (word | tag)) that HMMs require by using Bayesian inversion (i.e., using Bayes rule)

$$p(w_i|t_i) \quad = \quad P(t_i) * P(t_i|l_{n-i+1} \ldots l_n)).$$

Tagging Problem

Problem: for an input $x_1 \ldots x_n$, find

$$\arg \max_{y_1 \ldots y_{n+1}} p(x_1 \ldots x_n, y_1 \ldots y_{n+1})$$

where the $\arg\max$ is taken over all sequences $y_1 \ldots y_{n+1}$ such that $y_i \in \mathcal{S}$ for $i = 1 \ldots n$, and $y_{n+1} = \text{STOP}$.

We assume that p again takes the form

$$p(x_1 \ldots x_n, y_1 \ldots y_{n+1}) = \prod_{i=1}^{n+1} q(y_i | y_{i-2}, y_{i-1}) \prod_{i=1}^{n} e(x_i | y_i)$$

Recall that we have assumed in this definition that $y_0 = y_{-1} = *$, and $y_{n+1} = \text{STOP}$.

Brute Force Search is Hopelessly Inefficient

Problem: for an input $x_1 \ldots x_n$, find

$$\arg \max_{y_1 \ldots y_{n+1}} p(x_1 \ldots x_n, y_1 \ldots y_{n+1})$$

where the $\arg\max$ is taken over all sequences $y_1 \ldots y_{n+1}$ such that $y_i \in \mathcal{S}$ for $i = 1 \ldots n$, and $y_{n+1} = \text{STOP}$.

Let $|S| = 50$, length of sequence = n = 15

$|S|^n = 50^{15}$

Reading

- Chapter 8, Speech and Language Processing, Third Edition