Natural Language Processing

Lecture 9: Hidden Markov Models

Finding POS Tags

Bill directed plays about English kings

Running Example

Bill	directed	plays	about	English	kings
PropN	Adj	Verb	Prep	Adj	PlN
Verb	Verb	PlN	Adv	Noun	Verb
Noun			Part		

Running Example

Bill directed plays about English kings

Bill	directed	plays	about	English	kings
PropN	Adj	Verb	Prep	Adj	PlN
Verb	Verb	PlN	Adv	Noun	Verb
Noun			Part		

		p(t |Bill)
PropN	41	0.118
Verb	2	0.006
Noun	303	0.870

		p(t|directed)
Adj	0	0.000
Verb	10	1.000

		p(t|plays)
Verb	18	0.750
PlN	6	0.250

		p(t|about)
Prep	1546	0.750
Adv	502	0.244
Part	12	0.006

Running Example: POS

Bill directed plays about English kings

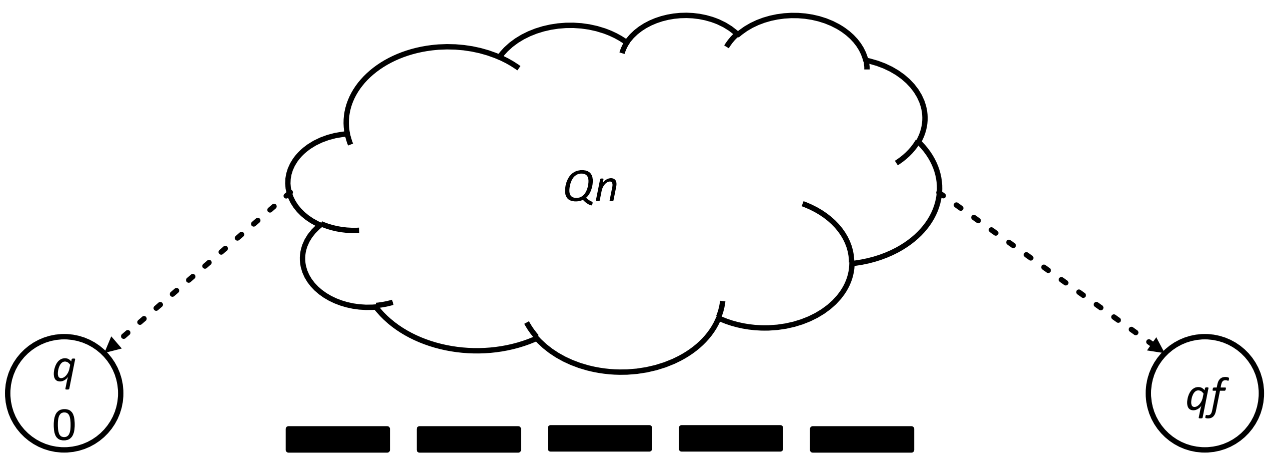
PropN	Adj	Verb	Prep	Adj	PlN
Verb	Verb	PlN	Adv	Noun	Verb
Noun			Part		

		p(t |English)
Adj	11	0.344
Noun	21	0.656

		p(t |kings)
PlN	3	1.000
Verb	0	0.000

Hidden Markov Model

- q_0: start state ("silent")
- q_f: final state ("silent")
- Q: set of "normal" states (excludes q_0 and final q_f)
- Σ: vocabulary of observable symbols
- $\gamma_{i,j}$: probability of transitioning to q_j given current state q_i
- $\eta_{i,w}$: probability of emitting $w \in \Sigma$ given current state q_i



HMM as a Noisy Channel

$p(x \mid y)$ using $\{\eta_{i,w}\}$

$p(y)$ using $\{\gamma_{i,j}\}$

source → y (tags) →  → x (words)

channel

decode

States vs. Tags

Running Example (prior)

Bill directed plays about English kings

PropN	Adj	Verb	Prep	Adj	PlN
Verb	Verb	PlN	Adv	Noun	Verb
Noun			Part		

p(PropN | <S> <S>)	0.202
p(Verb | <S> <S>)	0.023
p(Noun | <S> <S>)	0.040

Running Example

Bill directed plays about English kings

PropN	Adj	Verb	Prep	Adj	PlN
Verb	Verb	PlN	Adv	Noun	Verb
Noun			Part		

p(PropN | <S> <S>)	0.202	p(Adj | <S> PropN)	0.004	0.00081
		p(Verb | <S> PropN)	0.139	0.02808
p(Verb | <S> <S>)	0.023	p(Adj | <S> Verb)	0.062	0.00143
		p(Verb | <S> Verb)	0.032	0.00074
p(Noun | <S> <S>)	0.040	p(Adj | <S> Noun)	0.005	0.00020
		p(Verb | <S> Noun)	0.222	0.00888

Running Example

Bill directed plays about English kings

PropN	Adj	Verb	Prep	Adj	PlN
Verb	Verb	PlN	Adv	Noun	Verb
Noun			Part		

p(Adj | <S> PropN)	0.00081	p(Verb | PropN Adj)	0.011	0.00001
		p(PlN | PropN Adj)	0.157	0.00013
p(Verb | <S> PropN)	0.02808	p(Verb | PropN Verb)	0.162	**0.00455**
		p(PlN | PropN Verb)	0.022	0.00062
p(Adj | <S> Verb)	0.00143	p(Verb | Verb Adj)	0.009	0.00001
		p(PlN | Verb Adj)	0.246	0.00035
p(Verb | <S> Verb)	0.00074	p(Verb | Verb Verb)	0.078	0.00006
		p(PlN | Verb Verb)	0.034	0.00003
p(Adj | <S> Noun)	0.00020	p(Verb | Noun Adj)	0.020	0.00000
		p(PlN | Noun Adj)	0.103	0.00002
p(Verb | <S> Noun)	0.00888	p(Verb | Noun Verb)	0.176	0.00156
		p(PlN | Noun Verb)	0.018	0.00016

Running Example (posterior)

Bill directed plays about English kings

PropN Verb Noun	Adj Verb	Verb PlN	Prep Adv Part	Adj Noun	PlN Verb

		p(t |Bill)	p(Bill | t)
PropN	41	0.118	0.00044
Verb	2	0.006	0.00002
Noun	303	0.870	0.00228

Running Example

Bill directed plays about English kings

PropN	Adj	Verb	Prep	Adj	PlN
Verb	Verb	PlN	Adv	Noun	Verb
Noun			Part		

		p(t |directed)	p(directed |t)
Adj	0	0.000	0.00000
Verb	10	1.000	0.00008

Running Example

Bill directed plays about English kings

PropN	Adj	Verb	Prep	Adj	PlN
Verb	Verb	PlN	Adv	Noun	Verb
Noun			Part		

		p(t |plays)	p(plays |t)
Verb	18	0.750	0.00014
PlN	6	0.250	0.00010

Combining Two Components

- Prior $p(Y)$ the “language model”
  - What is the likelihood of a tag sequence
- Posterior $p(x|y)$ the “observation”
  - What is likelihood of word given tag
- We want to find the max for both
  - Bayes Rule $p(Y|X) = p(Y)\,p(X|Y) / p(X)$

HMM as a Noisy Channel

$p(x \mid y)$ using $\{\eta i,w\}$

$p(y)$ using $\{\gamma i,j\}$

source $\longrightarrow$ y (tags) $\longrightarrow$

$\longrightarrow$ x (words)

channel

decode

Part-of-Speech Tagging Task

- Input: a sequence of word tokens ***x***
- Output: a sequence of part-of-speech tags ***y***, one per word

HMM solution: find the most likely tag sequence, given the word sequence.

If I knew the best state sequence for words $x1 \ldots xn - 1$, then I could figure out the last state.

That decision would depend only on state $n - 1$.

$$\begin{aligned} y_n^* &= \arg\max_{q_i \in Q} p(Y_1 = y_1^*, \ldots, Y_{n-1} = y_{n-1}^*, Y_n = q_i \mid \boldsymbol{x}) \\ &= \arg\max_{q_i \in Q} V[n-1, y_{n-1}^*] \cdot \gamma_{y_{n-1}^*, i} \cdot \eta_{i, x_n} \cdot \gamma_{i,f} \\ &= \arg\max_{q_i \in Q} \gamma_{y_{n-1}^*, i} \cdot \eta_{i, x_n} \cdot \gamma_{i,f} \end{aligned}$$

I don't know that best sequence, but there are only $|Q|$ options at $n - 1$.

So I only need the score of the best sequence up to $n - 1$, ending in each possible state at $n - 1$. Call this $V[n - 1, q]$ for $q \in Q$.

Ditto, at every other timestep $n - 2$, $n - 3$, ... 1.

Viterbi Algorithm (Recursive Equations)

$$V[0, q_0] = 1$$

$$V[t, q_j] = \max_{q_i \in Q \cup \{q_0\}} V[t-1, q_i] \cdot \gamma_{i,j} \cdot \eta_{j,x_t}$$

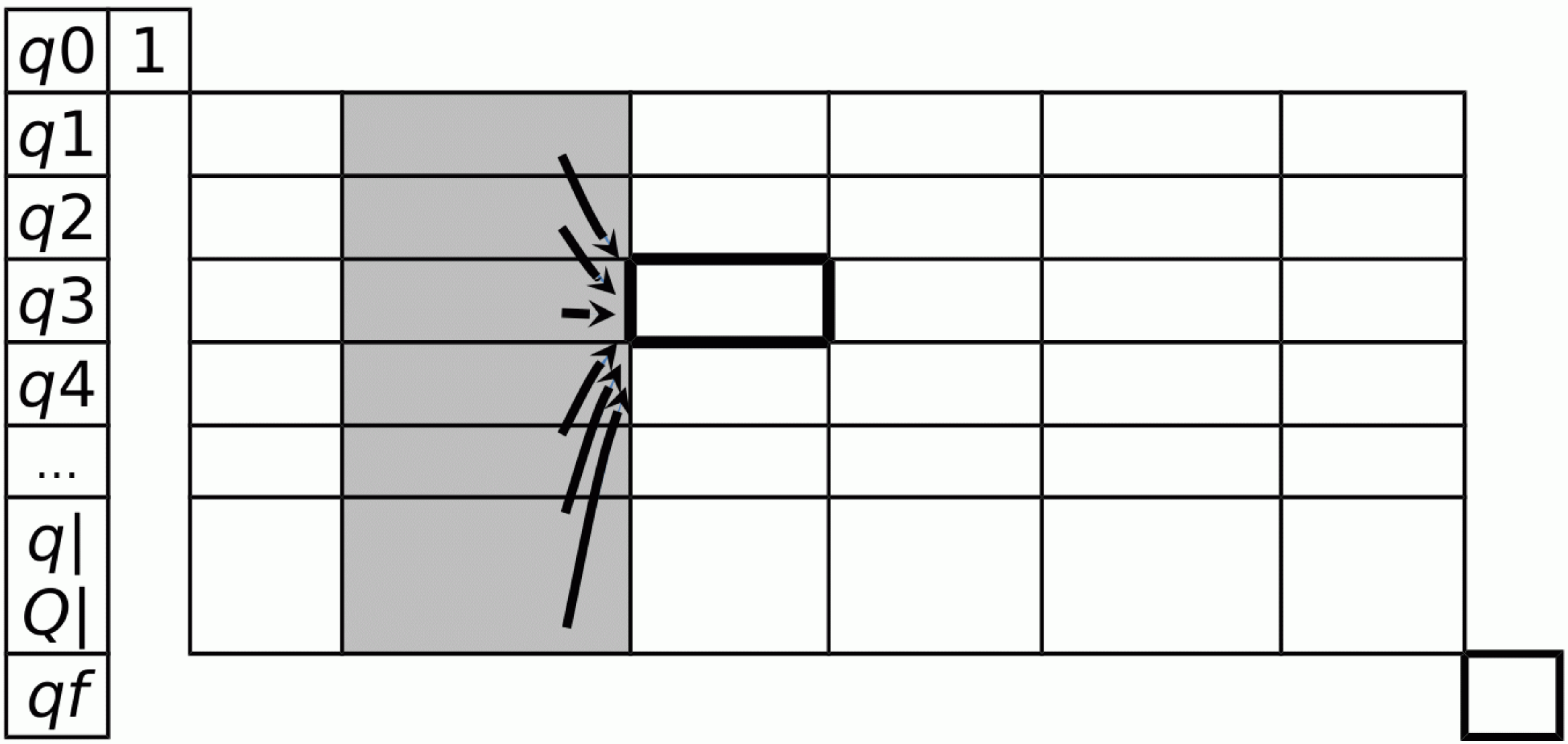
$$\mathrm{goal} = \max_{q_i \in Q} V[n, q_i] \cdot \gamma_{i,f}$$

Viterbi Algorithm (Procedure)

$V[*, *] \leftarrow 0$
$V[0, q0] \leftarrow 1$
for $t = 1 \ldots n$
  foreach qj
    foreach qi
      $V[t, qj] \leftarrow \max\{V[t, qj] , V[t - 1, qi] \times \gamma i,j \times \eta i,xt\}$
foreach qi
  goal $\leftarrow \max\{$ goal, $V[n, qi] \times \gamma i,f \}$
return goal

Running Example

Bill directed plays about English kings



Unknown words

- What is the PoS distribution of OOVs
  - Assume overall distribution from corpora
  - (Though less likely to be a Det, Conj, than Noun)
- Looking at the letters
  - Starts with a capital letter
  - Contains a number
  - Ends in “ed” or “ing”

Part of Speech in other Languages

- Need labeled data
  - Can be approximate, then correct it
- Morphologically rich languages
  - Need to decompose tokens to morphemes
  - Partly easier (but still PoS ambiguities)

Unsupervised PoS Tagging

- Words in the same context are the same Tag
  - Find all contexts: w1 X w2
  - Find most frequent Xs make them a tag
  - Repeat until you want to stop
- For English: do this 20 times
  - BE/HAVE MR/MRS AND/BUT/AT/AS
  - TO/FOR/OF/IN VERY/SO SHE/HE/IT/I/YOU
  - But no Nouns/Verb/Adj distinctions

Brown Clustering

- Unsupervised Word Clustering
- Non-syntax derived clusters
- "Semantically" related classes
- For example in a database of Flight information
  - To Shanghai, To Beijing, To London
  - To CLASS13, To CLASS13, To CLASS13
- Brown Clustering:
  - hierarchical agglomerative cluster.
  - Gives a binary tree, so it can easily scaled

Part of Speech and Tagging

- Reduced set of linguistic tags
  - Closed Class: Determiners, Pronouns …
  - Open Class: Nouns, Verbs, Adjs, Adverbs
- Probabilistic Labeling
  - Bayes/Noisy Channel
  - P(tag|word) * P(tag)
- HMMs, Viterbi decoding
- Unsupervised tagging/clustering
- Use what is *best* for your task
  - (and use what is available)