Part of Speech Tagging

Informatics 2A: Lecture 15

Mirella Lapata

School of Informatics
University of Edinburgh

21 October 2011

Benefits of Part of Speech Tagging

- **Can help in determining authorship**. Are any two documents written by the same person $\Rightarrow$ forensic linguistics.
- **Can help in speech synthesis and recognition**. For example, say the following **out-loud**

1. *Have you read 'The Wind in the Willows'?* (noun)
2. *The clock has stopped. Please wind it up.* (verb)
3. *The students tried to protest.* (verb)
4. *The students are pleased that their protest was successful.* (noun)

Corpus Annotation

Annotation: adds information that is not explicit in a corpus, increases its usefulness (often application-specific).

To annotate a coprus with Part-of-Speech (POS) classes we must define a tag set – the inventory of labels for marking up a corpus.

Example: part of speech tag sets

1. CLAWS tag (used for BNC); 62 tags;
2. Brown tag (used for Brown corpus); 87 tags;
3. Penn tag set (used for the Penn Treebank); 45 tags.

POS Tag Sets for English

Category	Examples	CLAWS	Brown	Penn
Adjective	happy, bad	AJ0	JJ	JJ
Noun singular	woman, book	NN1	NN	NN
Noun plural	women, books	NN2	NN	NN
Noun proper singular	London, Michael	NP0	NP	NNP
Noun proper plural	Finns, Hearts	NP0	NPS	NNPS
reflexive pro	itself, ourselves	PNX		
plural reflexive pro	ourselves, ...		PPLS	
Verb past participle	given, found	VVN	VBN	VBN
Verb base form	give, make	VVB	VB	VB
Verb simple past	ate, gave	VVD	VBD	VBD

All words must be assigned at least one tag. Differences in tags reflects what distinctions are/aren't drawn.

POS Tag Sets for English

Category	Examples	CLAWS	Brown	Penn
Adjective	happy, bad	AJ0	JJ	JJ
Noun singular	woman, book	NN1	NN	NN
Noun plural	women, books	NN2	NN	NN
Noun proper singular	London, Michael	NP0	NP	NNP
Noun proper plural	Finns, Hearts	NP0	NPS	NNPS
reflexive pro	itself, ourselves	PNX		
plural reflexive pro	ourselves, ...		PPLS	
Verb past participle	given, found	VVN	VBN	VBN
Verb base form	give, make	VVB	VB	VB
Verb simple past	ate, gave	VVD	VBD	VBD

All words must be assigned at least one tag. Differences in tags reflects what distinctions are/aren't drawn.

Tags and Tokens

In POS-tagged corpora tokens and their POS-tags are usually given in the form `text/tag`:

```
Our/PRP\$ enemies/NNS are/VBP innovative/JJ and/CC
resourceful/JJ ,/, and/CC so/RB are/VB we/PRP ./.
They/PRP never/RB stop/VB thinking/VBG about/IN new/JJ
ways/NNS to/TO harm/VB our/PRP\$ country/NN and/CC
our/PRP\$ people/NN, and/CC neither/DT do/VB we/PRP
```

Extent of POS Ambiguity

- POS-tagging a large corpus by hand is a lot of work.
- We'd prefer to automate but how hard can it be?
- Many words may appear in several categories.
- But most words appear most of the time in one category.

POS Ambiguity in the Brown corpus

Brown corpus (1M words) has 39,440 different word types:

- 35340 have only 1 POS tag anywhere in corpus (89.6%)
- 4100 (10.4%) have 2–7 POS tags

Why does 10.4% POS-tag ambiguity by word type lead to difficulty?

Extent of POS Ambiguity

- Words in a large corpus have a Zipfian distribution.
- Many high frequency words have more than one POS tag.
- More than 40% of the word tokens are ambiguous.

He wants **to/TO** go.
He went **to/IN** the store.

He wants **that/DT** hat.
It is obvious **that/CS** he wants a hat.
He wants a hat **that/WPS** fits.

Extent of POS Ambiguity

- Words in a large corpus have a Zipfian distribution.
- Many high frequency words have more than one POS tag.
- More than 40% of the word tokens are ambiguous.

He wants **to/TO** go.
He went **to/IN** the store.

He wants **that/DT** hat.
It is obvious **that/CS** he wants a hat.
He wants a hat **that/WPS** fits.

How about guessing the most common tag for each word?

Extent of POS Ambiguity

- Words in a large corpus have a Zipfian distribution.
- Many high frequency words have more than one POS tag.
- More than 40% of the word tokens are ambiguous.

He wants **to/TO** go.
He went **to/IN** the store.

He wants **that/DT** hat.
It is obvious **that/CS** he wants a hat.
He wants a hat **that/WPS** fits.

How about guessing the most common tag for each word?
Will give you 90% accuracy (state of-the-art is 96–98%).

Clicker Question

What is the difference between word types and tokens?

1. Word types are part of speech tags, tokens are just the words.
2. Word types are the number of times words appear in the corpus, whereas word tokens are unique occurrences of words in the corpus.
3. Word types are the vocabulary (what different words are there), whereas word tokens refer to the frequency of each word type.
4. Word types and tokens are the same thing.

Sequence Labeling

Find the best sequence of tags that corresponds to:

Secrertariat	is	expected	to	race	tomorrow
NNP	VBZ	VBN	TO	VB	NN
NNP	VBZ	VBN	TO	NN	NN

Sequence Labeling

Find the best sequence of tags that corresponds to:

Secrertariat	is	expected	to	race	tomorrow
NNP	VBZ	VBN	TO	VB	NN
NNP	VBZ	VBN	TO	NN	NN

Sequence Labeling

Find the best sequence of tags that corresponds to:

Secrertariat	is	expected	to	race	tomorrow
NNP	VBZ	VBN	TO	VB	NN
NNP	VBZ	VBN	TO	NN	NN

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

Sequence Labeling

Find the best sequence of tags that corresponds to:

Secrertariat	is	expected	to	race	tomorrow
NNP	VBZ	VBN	TO	VB	NN
NNP	VBZ	VBN	TO	NN	NN

$$\begin{aligned}\hat{t}_1^n &= \operatorname*{argmax}_{t_1^n} P(t_1^n|w_1^n) \\ &= \operatorname*{argmax}_{t_1^n} \frac{P(w_1^n|t_1^n)P(t_1^n)}{P(w_1^n)} \quad \text{using Bayes' rule}\end{aligned}$$

Sequence Labeling

Find the best sequence of tags that corresponds to:

Secrertariat	is	expected	to	race	tomorrow
NNP	VBZ	VBN	TO	VB	NN
NNP	VBZ	VBN	TO	NN	NN

$$
\begin{aligned}
\hat{t}_1^n &= \operatorname*{argmax}_{t_1^n} P(t_1^n|w_1^n) \\
&= \operatorname*{argmax}_{t_1^n} \frac{P(w_1^n|t_1^n)P(t_1^n)}{P(w_1^n)} \quad \text{using Bayes' rule} \\
&= \operatorname*{argmax}_{t_1^n} P(w_1^n|t_1^n)P(t_1^n) \quad \text{denominator does not change}
\end{aligned}
$$

Sequence Labeling

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} \underbrace{P(w_1^n | t_1^n)}_{likelihood} \quad \underbrace{P(t_1^n)}_{prior}$$

Sequence Labeling

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

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

Sequence Labeling

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} \underbrace{P(w_1^n|t_1^n)}_{likelihood} \quad \underbrace{P(t_1^n)}_{prior}$$

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

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

Sequence Labeling

$$\hat{t}_1^n = \operatorname*{argmax}_{t_1^n} \underbrace{P(w_1^n|t_1^n)}_{\textit{likelihood}} \; \underbrace{P(t_1^n)}_{\textit{prior}} \approx \prod_{i=1}^{n} P(w_i|t_i) \prod_{i=1}^{n} P(t_i|t_{i-1})$$

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

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

Sequence Labeling

$$\hat{t}_1^n \approx \operatorname*{argmax}_{t_1^n} \underbrace{\prod_{i=1}^{n} P(w_i|t_i)}_{emission\ probability} \underbrace{\prod_{i=1}^{n} P(t_i|t_{i-1})}_{transition\ probability}$$

Sequence Labeling

$$\hat{t_1^n} \approx \underset{t_1^n}{\operatorname{argmax}} \underbrace{\prod_{i=1}^{n} P(w_i|t_i)}_{emission\ probability} \underbrace{\prod_{i=1}^{n} P(t_i|t_{i-1})}_{transition\ probability}$$

$P(w_i|t_i) = \frac{C(t_i,w_i)}{C(t_i)}$

Sequence Labeling

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

$$P(w_i|t_i) = \frac{C(t_i, w_i)}{C(t_i)}$$

$$P(t_i|t_{i-1}) = \frac{C(t_i, t_{i-1})}{C(t_{i-1})}$$

Sequence Labeling

$$\hat{t}_1^n \approx \underset{t_1^n}{\operatorname{argmax}} \underbrace{\prod_{i=1}^{n} P(w_i|t_i)}_{\textit{emission probability}} \underbrace{\prod_{i=1}^{n} P(t_i|t_{i-1})}_{\textit{transition probability}}$$

$$P(w_i|t_i) = \frac{C(t_i, w_i)}{C(t_i)}$$

$$P(t_i|t_{i-1}) = \frac{C(t_i, t_{i-1})}{C(t_{i-1})}$$

$$P(is|VBZ) = \frac{C(VBZ, is)}{C(VBZ)} = \frac{10{,}073}{21{,}627} = .47$$

$$P(NN|DT) = \frac{C(DT, NN)}{C(DT)} = \frac{56{,}509}{116{,}454} = .49$$

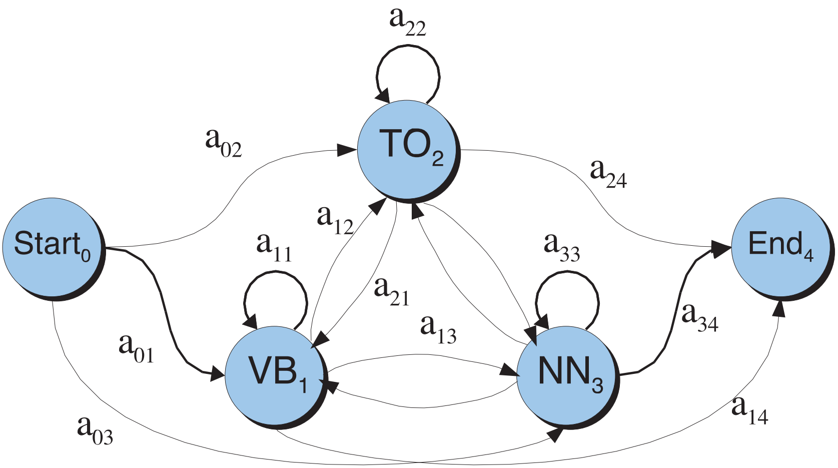
Hidden Markov Models

- A **finite automaton** is defined by set of states and set of transitions between states according to input observations
- A **weighted finite automaton** has probabilities or weights on the arcs
- In a **Markov chain** the input sequence uniquely determines which states the automaton will go through.
- In a **Hidden Markov model** the sequence of states given input is hidden, i.e., ambiguous.
- In POS-tagging, we observe the input words but not the POS-tags themselves.

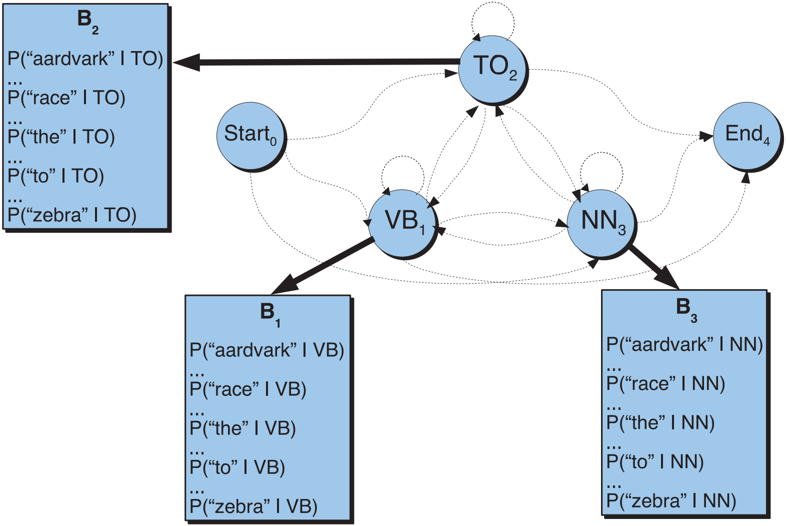
Definition of Hidden Markov Models

$Q = q_1, q_2 \ldots q_N$	A set of N **states**
$A = a_{11} a_{12} \ldots a_{n1} \ldots a_{nn}$	a **transition probability matrix** A, each a_{ij} represents the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$O = o_1, o_2 \ldots o_T$	sequence of T **observations** drawn from vocabulary $V = v_1, v_2 \ldots v_V$.
$B = b_i(o_T)$	Sequence of **emission probabilities** expressing probability of o_t being generated from state i.
q_0, q_F	a **start state** and **final state**.

Transition Probabilities



Emission Probabilities



Transition and Emission Probabilities

	VB	**TO**	**NN**	**PPPS**
<s>	.019	.0043	.041	.67
VB	.0038	.035	.047	.0070
TO	.83	0	.000	0
NN	.0040	.016	.087	.0045
PPPS	.23	.00079	.001	.00014

	I	**want**	**to**	**race**
VB	0	.0093	0	.00012
TO	0	0	.99	0
BB	0	.000054	0	.00057
PPSS	.37	0	0	0

How Do we Search for Best Tag Sequence?

We have defined an HMM, but how do we use it? We are given a **word sequence** and must find their corresponding **tag sequence**.

- It is easy to compute the probability of a specific tag sequence:

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

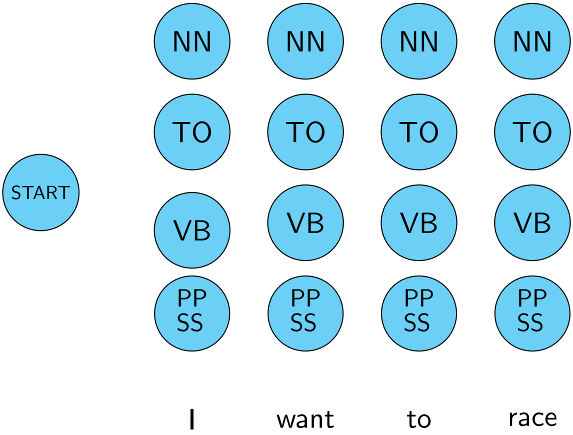
- But how do we find most likely tag sequence?
- We can do this efficiently using **dynamic programming** and the **Viterbi algorithm**.

Clicker Question

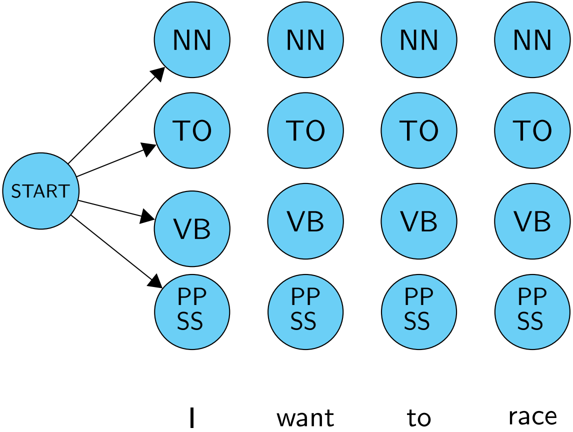
Given n words and on average T choices, how many tag sequences do we have to evaluate?

1. $|T|$ tag sequences
2. n tag sequences
3. $|T| \times n$ tag sequences
4. $|T|^n$ tag sequences

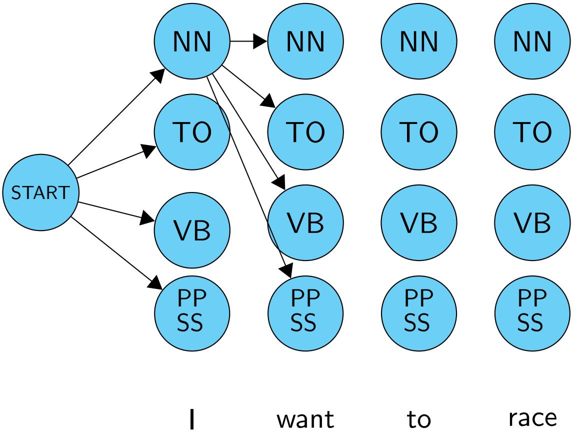
The HMM Trellis



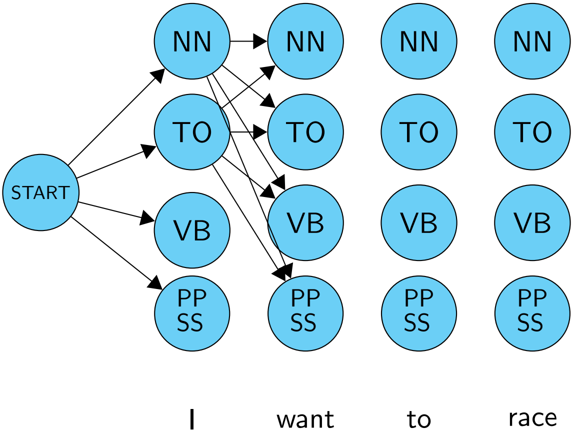
The HMM Trellis



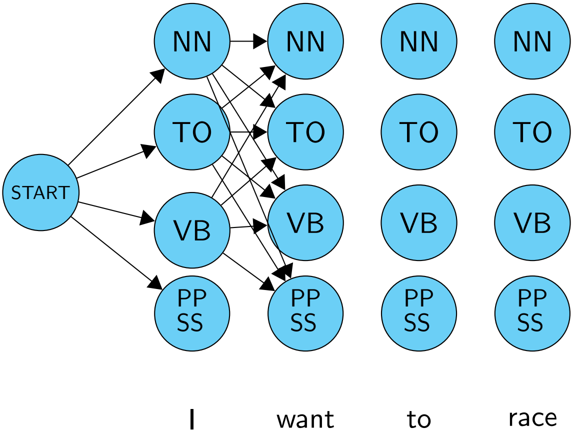
The HMM Trellis



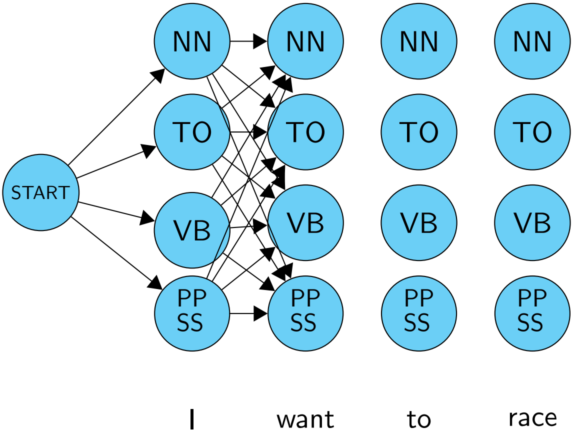
The HMM Trellis



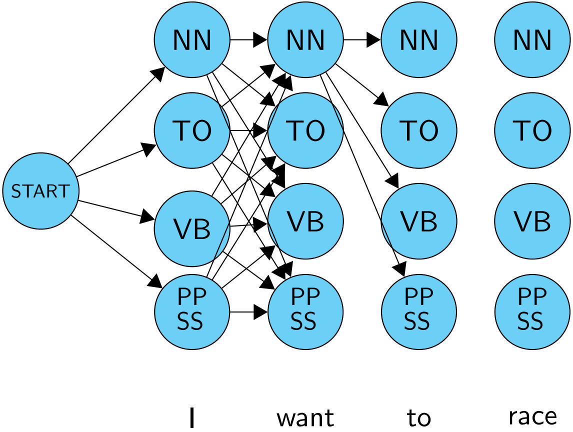
The HMM Trellis



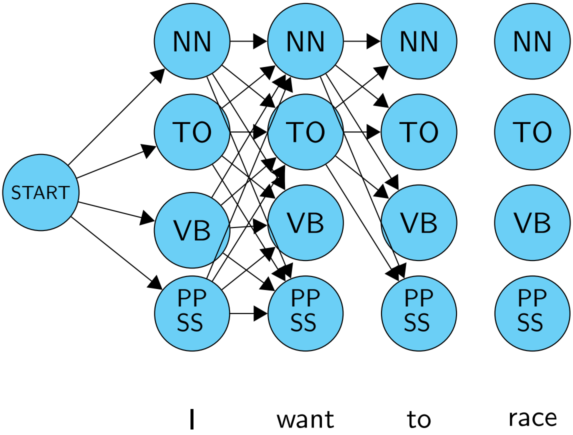
The HMM Trellis



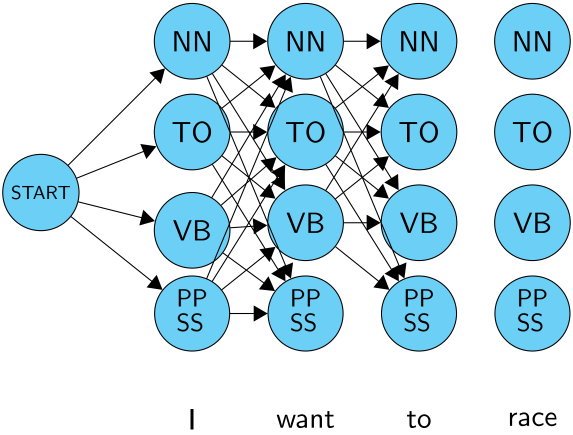
The HMM Trellis



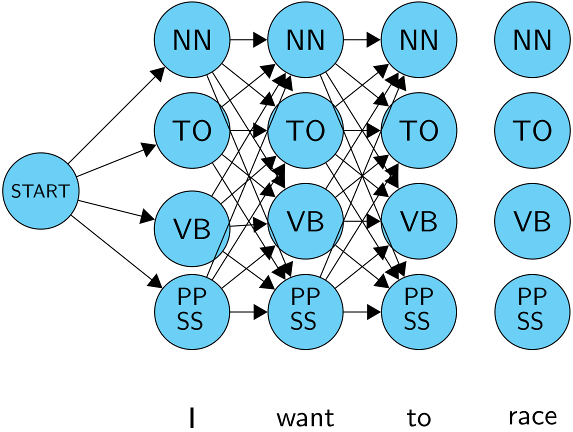
The HMM Trellis



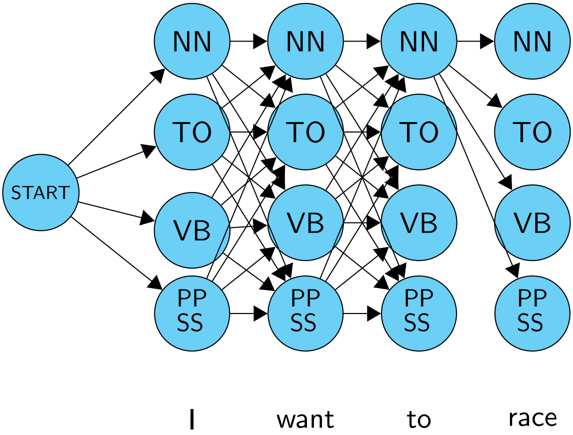
The HMM Trellis



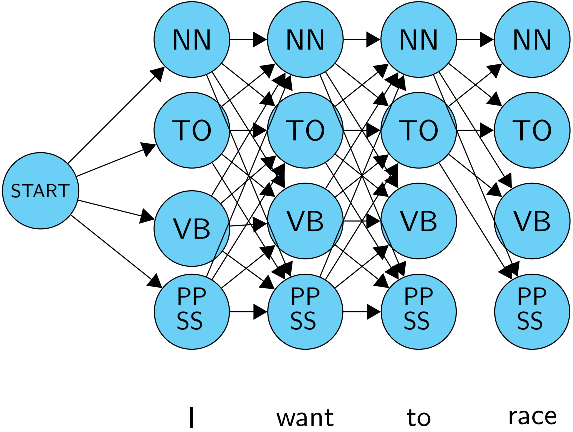
The HMM Trellis



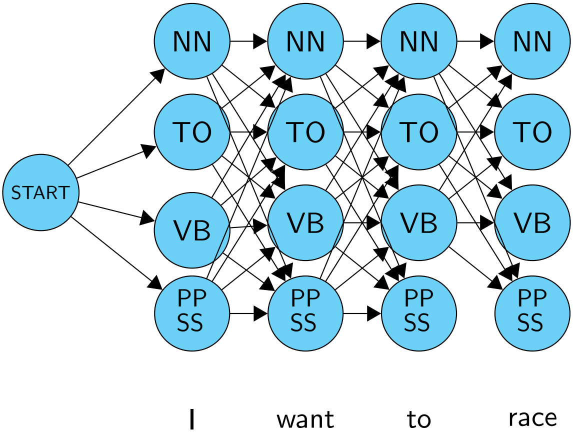
The HMM Trellis



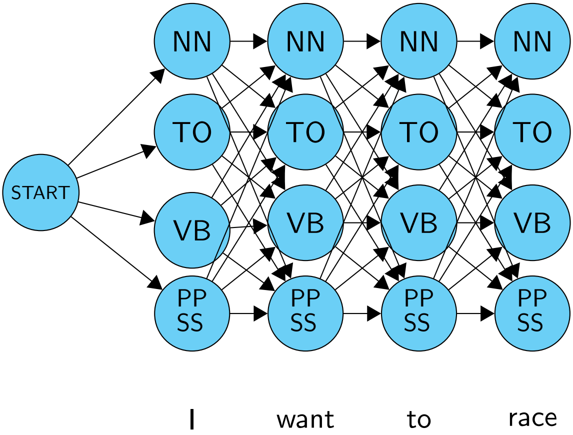
The HMM Trellis



The HMM Trellis



The HMM Trellis



The Viterbi Algorithm

q_{end}	end					
q_4	NN	0				
q_3	TO	0				
q_2	VB	0				
q_1	PPSS	0				
q_o	start	**1.0**				
		<s>	I	want	to	race
		o_o	o_1	o_2	o_3	o_4

1. Create probability matrix, with one column for each observation (i.e., word), and one row for each state (i.e., tag).
2. We proceed by filling cells, column by column

The Viterbi Algorithm

q_{end}	end					
q_4	NN	0	$1.0 \times .041 \times 0$			
q_3	TO	0	$1.0 \times .0043 \times 0$			
q_2	VB	0	$1.0 \times .19 \times 0$			
q_1	PPSS	0	$1.0 \times .67 \times .37$			
q_o	start	**1.0**				
		<s>	I	want	to	race
		o_o	o_1	o_2	o_3	o_4

- For each state q_j at time t compute

  $$v_t(j) = \max_{i=j}^{N} v_{t-1}(i) a_{ij} b_j(o_t)$$

- $v_{t-1}(i)$ is **previous Viterbi path probability**, a_{ij} is **transition probability**, and $b_j(o_t)$ is **emission probability**

The Viterbi Algorithm

q_{end}	end					
q_4	NN	0	0	$.025 \times .0012 \times 0.000054$		
q_3	TO	0	0	$.025 \times .00079 \times 0$		
q_2	VB	0	0	$.025 \times .23 \times .0093$		
q_1	PPSS	0	**.025**	$.025 \times .00014 \times 0$		
q_0	start	**1.0**				
		<s>	I	want	to	race
		o_o	o_1	o_2	o_3	o_4

- For each state q_j at time t compute
  $$v_t(j) = \max_{i=j}^{N} v_{t-1}(i) a_{ij} b_j(o_t)$$
- $v_{t-1}(i)$ is **previous Viterbi path probability**, a_{ij} is **transition probability**, and $b_j(o_t)$ is **state observation likelihood**

The Viterbi Algorithm

q_{end}	end					
q_4	NN	0	0	.000000002	.000053 × .047 × 0	
q_3	TO	0	0	0	.000053 × .035 × .99	
q_2	VB	0	0	**.00053**	.000053 × .0038 × 0	
q_1	PPSS	0	**.025**	0	.000053 × .0070 × 0	
q_0	start	**1.0**				
		<s>	I	want	to	race
		o_o	o_1	o_2	o_3	o_4

- For each state q_j at time t compute

  $$v_t(j) = \max_{i=j}^{N} v_{t-1}(i) a_{ij} b_j(o_t)$$

- $v_{t-1}(i)$ is **previous Viterbi path probability**, a_{ij} is **transition probability**, and $b_j(o_t)$ is **state observation likelihood**

The Viterbi Algorithm

q_{end}	end					
q_4	NN	0	0	.000000002	0	$.0000018 \times .00047 \times .00057$
q_3	TO	0	0	0	**.0000018**	$.0000018 \times 0 \times 0$
q_2	VB	0	0	**.00053**	0	$.0000018 \times .83 \times .00012$
q_1	PPSS	0	**.025**	0	0	$.0000018 \times 0 \times 0$
q_0	start	**1.0**				
		<s>	I	want	to	race
		o_o	o_1	o_2	o_3	o_4

- For each state q_j at time t compute
  $v_t(j) = \max_{i=j}^{N} v_{t-1}(i) a_{ij} b_j(o_t)$
- $v_{t-1}(i)$ is **previous Viterbi path probability**, a_{ij} is **transition probability**, and $b_j(o_t)$ is **state observation likelihood**

The Viterbi Algorithm

q_{end}	end						
q_4	NN	0	0	.000000002	0	4.8222e-13
q_3	TO	0	0	0	**.0000018**	0
q_2	VB	0	0	**.00053**	0	**1.7928e-10**
q_1	PPSS	0	**.025**	0	0	0
q_0	start	**1.0**				
		<s>	I	want	to	race
		o_o	o_1	o_2	o_3	o_4

- For each state q_j at time t compute
  $$v_t(j) = \max_{i=j}^{N} v_{t-1}(i) a_{ij} b_j(o_t)$$
- $v_{t-1}(i)$ is **previous Viterbi path probability**, a_{ij} is **transition probability**, and $b_j(o_t)$ is **state observation likelihood**

Summary

- A number of POS tag sets exist for English (e.g. Brown, CLAWS, Penn).
- Automatic POS tagging makes errors because many high frequency words are part-of-speech ambiguous.
- POS-tagging can be performed automatically using Hidden Markov Models.

Reading:	J&M (2nd edition) Chapter 5 NLTK Book: Chapter 5, Categorizing and Tagging Words
Next lecture:	Phrase structure and parsing as search