Text Mining

February 28 2019

Part-of-Speech (POS) Tagging

- Symbolic
  - Rule-based
  - Transformation-based
- Probabilistic
  - Hidden Markov Models
  - Maximum Entropy Markov Models
  - Conditional Random Fields

Part-of-Speech Tagging (POS)

- Task of tagging POS tags (Nouns, Verbs, Adjectives, Adverbs, ...) for words
- POS tags provide lot of information about a word
  - knowing whether a word is **noun** or **verb** gives information about neighbouring words
  - nouns are preceded by determiners; adjectives and verbs by nouns
  - useful for Named entity recognition; Machine Translation; Parsing; Word sense disambiguation
- Given a word, we assume it can belong to only one of the POS tags.
- POS Tagging problem
  - Given a sentence $S = w_1 w_2 .... w_n$ consisting of n words, determine the corresponding tag sequence $P = P_1 P_2 .... P_n$

POS Tagging - Challenges

- Words often have more than one POS: e.g., back
  - *The back door* = adjective (JJ)
  - *On my back* = noun (NN)
  - *Win the voters back* = adverb (RB)
  - *Promised to back the bill* = verb (VB)

POS Tagging - Tagset

Tag	Description	Example	Tag	Description	Example
CC	coordin. conjunction	*and, but, or*	SYM	symbol	+,%, &
CD	cardinal number	*one, two*	TO	“to”	*to*
DT	determiner	*a, the*	UH	interjection	*ah, oops*
EX	existential ‘there’	*there*	VB	verb base form	*eat*
FW	foreign word	*mea culpa*	VBD	verb past tense	*ate*
IN	preposition/sub-conj	*of, in, by*	VBG	verb gerund	*eating*
JJ	adjective	*yellow*	VBN	verb past participle	*eaten*
JJR	adj., comparative	*bigger*	VBP	verb non-3sg pres	*eat*
JJS	adj., superlative	*wildest*	VBZ	verb 3sg pres	*eats*
LS	list item marker	*1, 2, One*	WDT	wh-determiner	*which, that*
MD	modal	*can, should*	WP	wh-pronoun	*what, who*
NN	noun, sing. or mass	*llama*	WP$	possessive wh-	*whose*
NNS	noun, plural	*llamas*	WRB	wh-adverb	*how, where*
NNP	proper noun, sing.	*IBM*	$	dollar sign	*$*
NNPS	proper noun, plural	*Carolinas*	#	pound sign	*#*
PDT	predeterminer	*all, both*	“	left quote	‘ or “
POS	possessive ending	*’s*	”	right quote	’ or ”
PRP	personal pronoun	*I, you, he*	(	left parenthesis	[, (, {, <
PRP$	possessive pronoun	*your, one’s*	)	right parenthesis	],), }, >
RB	adverb	*quickly, never*	,	comma	,
RBR	adverb, comparative	*faster*	.	sentence-final punc	. ! ?
RBS	adverb, superlative	*fastest*	:	mid-sentence punc	: ; ... – -
RP	particle	*up, off*			

Figure: Penn Treebank POS Tags

POS Tagging - Brown Corpus

- **Brown Corpus** - standard corpus used for POS tagging task
- first text corpus of American English
- published in 1963-1964 by Francis and Kucera
- consists of 1 million words (500 samples of 2000+ words each)
- Brown corpus is PoS tagged with Penn TreeBank tagset.
- $\approx$ 11% of the word types are ambiguous with regard to POS
- $\approx$ 40% of the word tokens are ambiguous
- ambiguity for common words. e.g. **that**
  - I know **that** he is honest = preposition (IN)
  - Yes, **that** play was nice = determiner (DT)
  - You can't to **that** far = adverb (RB)

Automatic POS Tagging

- Symbolic
  - Rule-based
  - Transformation-based
- Probabilistic
  - Hidden Markov Models
  - Maximum Entropy Markov Models
  - Conditional Random Fields

Automatic POS Tagging - Brill Tagger

- An example of Transformation-Based Learning
  - Basic idea: do a quick job first (using frequency), then revise it using contextual rules.
  - Painting metaphor from the readings
- Very popular (freely available, works fairly well)
- A supervised method: requires a tagged corpus

Automatic POS Tagging - Brill Tagger

- Start with simple (less accurate) rules…learn better ones from tagged corpus
  - Tag each word initially with most likely POS
  - Examine set of transformations to see which improves tagging decisions compared to tagged corpus
  - Re-tag corpus using best transformation
  - Repeat until, e.g., performance doesn't improve
  - Result: tagging procedure (ordered list of transformations) which can be applied to new, untagged text

Automatic POS Tagging: Brill Tagger - Example

- Examples:
  - They are expected to race tomorrow.
  - The race for outer space.
- Tagging algorithm:
  1. Tag all uses of “race” as NN (most likely tag in the Brown corpus)
     - They are expected to race/NN tomorrow
     - the race/NN for outer space
  2. Use a transformation rule to replace the tag NN with VB for all uses of “race” preceded by the tag TO:
     - They are expected to race/VB tomorrow
     - the race/NN for outer space

Automatic POS Tagging: Brill Tagger - Sample Final Rules

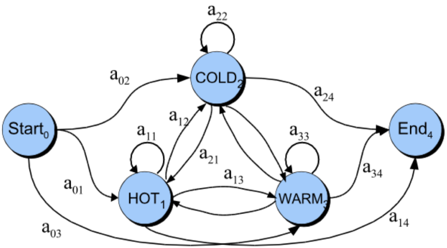
```
Rules:
NN -> NNP if the tag of words i+1...i+2 is 'NNP'
NN -> VB if the tag of the preceding word is 'TO'
NN -> VBD if the tag of the following word is 'DT'
NN -> VBD if the tag of the preceding word is 'NNS'
NN -> JJ if the tag of the preceding word is 'DT', and the tag of the followi
ng word is 'NN'
NN -> NNP if the tag of the preceding word is 'NN', and the tag of the follow
ing word is ','
NN -> NNP if the tag of words i+1...i+2 is 'NNP'
NN -> IN if the tag of the preceding word is '.'
NNP -> NN if the tag of words i-3...i-1 is 'JJ'
NN -> JJ if the tag of the following word is 'JJ'
NN -> VBP if the tag of the preceding word is 'PRP'
WDT -> IN if the tag of the following word is 'DT'
NN -> JJ if the tag of the preceding word is 'IN', and the tag of the followi
ng word is 'NN'
NN -> VBN if the tag of the preceding word is 'VBP'
VBD -> VB if the tag of the preceding word is 'MD'
NN -> JJ if the tag of the preceding word is 'CC', and the tag of the followi
ng word is 'NN'
```

Automatic POS Tagging

- Symbolic
  - Rule-based
  - Transformation-based
- Probabilistic
  - Hidden Markov Models (HMM)
  - Maximum Entropy Markov Models (MEMM)
  - Conditional Random Field (CRF)

Markov Chains

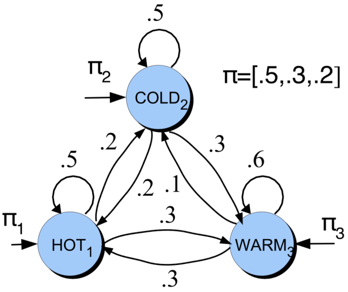
- Probabilistic **graphical model** for representing probabilistic assumptions in a graph.



- $Q = q_1, q_2, ..., q_N$: a set of **states**
- $A = a_{01}, a_{02}, ...a_{n1}, ..., a_{nn}$: a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, $s.t. \sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
- q_0, q_{end} : a special *start and end state* which are not associated with observations

Markov Chains

$\pi_1, \pi_2, ..., \pi_N$: an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i.



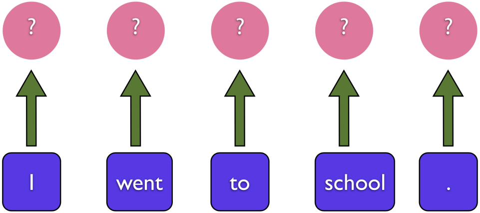
- Markov Assumption:
  $P(q_i|q_1, q_2, ..., q_{i-1}) = P(q_i|q_{i-1})$

- $P(cold\ hot\ cold\ hot) =$
  $P(cold)\ P(hot|cold)\ P(cold|hot)\ P(hot|cold)$
  $= 0.3 \times 0.2 \times 0.2 \times 0.2$
  $= 0.0024$

Hidden Markov Model (HMM)

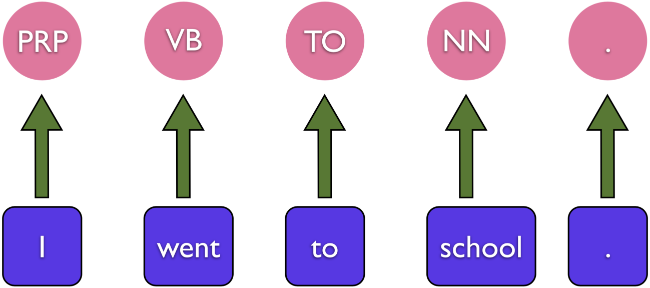
- Markov chains are useful for observed events
- However, in many cases the events are not observed
  - Example: POS tagging - POS tags are not observed

Given a sequence of words (observed states) determine a sequence of state transitions (unobserved states)



- HMMs allows us to model both *observed events* (words that we see) and *hidden events* (POS tags).

Hidden Markov Model (HMM)



HMM - Definition

$Q = q_1 q_2 \ldots q_N$	a set of **states**
$A = a_{01} a_{02} \ldots a_{n1} \ldots a_{nn}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \ldots o_N$	a set of **observations**, each one drawn from a vocabulary $V = v_1, v_2, ..., v_V$.
$B = b_i(o_t)$	A set of **observation likelihoods:**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state i.
q_0, q_{end}	a special **start and end state** which are not associated with observation.

Markov Assumption: $P(q-1|q_1, ...q_{i-1} = P(q_i|q_{i-1})$

Output Independence Assumption:

$$P(o_i|q_1, ..., q_i, ..., q_n, o_1, ..., o_i, ..., o_n) = P(o_i|q_i)$$

A motivating example

Urn 1
# of Red = 30
# of Green = 50
# of Blue = 20

Urn 2
# of Red = 10
# of Green = 40
# of Blue = 50

Urn 3
# of Red = 60
# of Green = 10
# of Blue = 30

Probability of transition to another Urn after picking a ball:

	U_1	U_2	U_3
U_1	0.1	0.4	0.5
U_2	0.6	0.2	0.2
U_3	0.3	0.4	0.3

A Motivating Example (contd.)

Given: Transition Probabilities

	U_1	U_2	U_3
U_1	0.1	0.4	0.5
U_2	0.6	0.2	0.2
U_3	0.3	0.4	0.3

Given: Output Probabilities

	R	G	B
U_1	0.3	0.5	0.2
U_2	0.1	0.4	0.5
U_3	0.6	0.1	0.3

Observation: RRGGBRGR

State Sequence (Urn chosen corresponding to each ball): ?

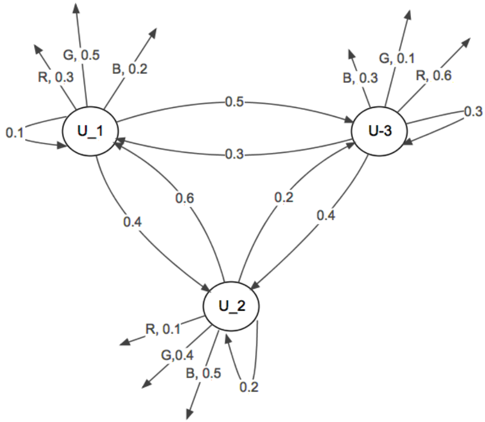
Diagrammatic Representation - 1

Transition Probabilities

	U_1	U_2	U_3
U_1	0.1	0.4	0.5
U_2	0.6	0.2	0.2
U_3	0.3	0.4	0.3

Output Probabilities

	R	G	B
U_1	0.3	0.5	0.2
U_2	0.1	0.4	0.5
U_3	0.6	0.1	0.3



Observation: RRGGBRGR

State Sequence (Urn chosen corresponding to each ball): ?

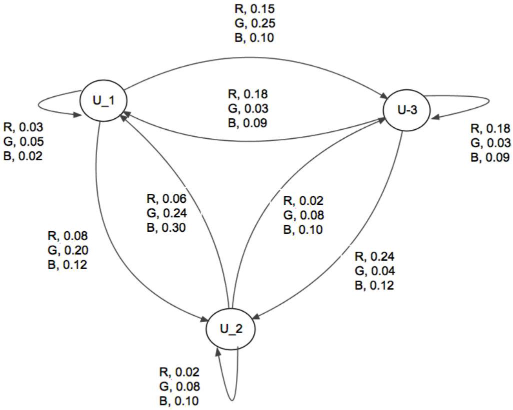
Diagrammatic Representation - 2

Transition Probabilities

	U_1	U_2	U_3
U_1	0.1	0.4	0.5
U_2	0.6	0.2	0.2
U_3	0.3	0.4	0.3

Output Probabilities

	R	G	B
U_1	0.3	0.5	0.2
U_2	0.1	0.4	0.5
U_3	0.6	0.1	0.3



Observation: RRGGBRGR

State Sequence (Urn chosen corresponding to each ball): ?

Example (contd.)

- States Set: $S = \{U_1, U_2, U_3\}$
- Observation Set: $V = \{R, G, B\}$
- Observation Sequence:
  - $O = \{O_1 \ldots . O_n\}$
- State Sequence:
  - $Q = \{q_1 \ldots . q_n\}$
- Initial Probability: ϵ
  - $\epsilon_i = P(q_i = U_i)$

Transition Probabilities (A)

	U_1	U_2	U_3
U_1	0.1	0.4	0.5
U_2	0.6	0.2	0.2
U_3	0.3	0.4	0.3

Output Probabilities (B)

	R	G	B
U_1	0.3	0.5	0.2
U_2	0.1	0.4	0.5
U_3	0.6	0.1	0.3

Observations and states

	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8
OBS:	R	R	G	G	B	R	G	R
State:	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8

$S_i = U_1/U_2/U_3$; A particular state
S: State sequence
O: Observation sequence
S* = 'best' possible state (urn) sequence
Goal: Maximize $P(S * |O$ by choosing 'best' S

- Goal: Maximize $P(S|O)$ where S is the State Sequence and O is the Observation Sequence
    - $S^* = argmax_s(P(S|O))$

	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8
OBS:	R	R	G	G	B	R	G	R
State:	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8

$P(S|O) = P(S_{1-8}|O_{1-8})$
$P(S|O) = P(S_1|O)P(S_2|S_1, O)P(S_3|S_{1-2}, O)....P(S_8|S_{1-7}, O)$

Markov Assumption: a state depends only on the previous state

$P(S|O) = P(S_1|O)P(S_2|S_1, O)P(S_3|S_2, O)....P(S_8|S_7, O)$

Baye's Theorem

$P(A|B) = \frac{P(A)P(B|A)}{P(B)}$
$P(A)$: Prior $P(B|A)$: Likelihood

$argmax_s P(S|O) = argmax_x P(S)P(O|S)$

State Transitions Probability

$P(S) = P(S_{1-8})$
$P(S) = P(S_1)P(S_2|S_1)P(S_3|S_{1-2})P(S_4|S_{1-3})...P(S_8|S_{1-7})$

By Markov Assumption (k=1)

$P(S) = P(S_1)P(S_2|S_1)P(S_3|S_2)P(S_4|S_3)....P(S_8|S_7)$

Observations Sequence Probability

$P(O|S) =$
$P(O_1|S_{1-8})P(O_2|O_1, S_{1-8})P(O_3|O_{1-2}, S_{1-8})...P(O_8|O_{1-7}, S_{1-8})$

Assumption that ball drawn depends only on the Urn Chosen

$P(O|S) = P(O_1|S_1)P(O_2|S_2)P(O_3|S_3)....P(O_8|S_8)$

$P(S|O) = P(S)P(O|S)$

$P(S|O) = P(S_1)P(S_2|S_1)P(S_3|S_2)P(S_4|S_3)....P(S_8|S_7)P(O_1|S_1)$
$P(O_2|S_2)P(O_3|S_3)....P(O_8|S_8)$

	O_0	O_1	0_2	O_3	O_4	O_5	O_6	O_7	O_8	
OBS:	ϵ	R	R	G	G	B	R	G	R	
State:	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9

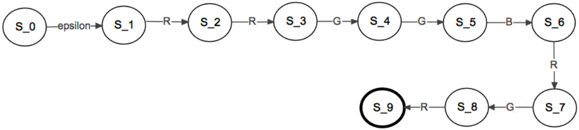
$$
\begin{aligned}
& P(S).P(O|S) \\
= \quad & [P(O_0|S_0).P(S_1|S_0)] \\
& [P(O_1|S_1).P(S_2|S_1)] \\
& [P(O_2|S_2).P(S_3|S_2)] \\
& [P(O_3|S_3).P(S_4|S_3)] \\
& [P(O_4|S_4).P(S_5|S_4)] \\
& [P(O_5|S_5).P(S_6|S_5)] \\
& [P(O_6|S_6).P(S_7|S_6)] \\
& [P(O_7|S_7).P(S_8|S_7)] \\
& [P(O_8|S_8).P(S_9|S_8)]
\end{aligned}
$$

States S_0 and S_9 is introduced as initial and final states

After S_8 the next state is S_9 with probability 1, i.e., $P(S_9|S_8) == 1$

O_0 is ϵ-transition

	O_0	O_1	0_2	O_3	O_4	O_5	O_6	O_7	O_8	
OBS:	ϵ	R	R	G	G	B	R	G	R	
State:	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9



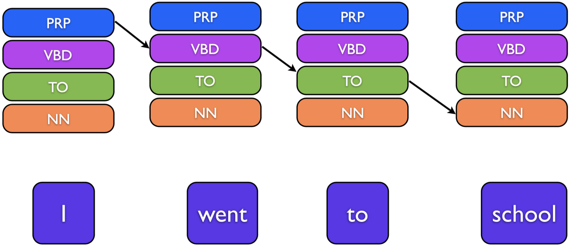
$$P(O_k|S_k).P(S_{k+1}|S_k) = P(S_k \xrightarrow{o_k} S_{k+1})$$

Three problems of HMM

- **Problem 1 (Decoding)**: Given an observation sequence O and an HMM $\lambda = (A, B)$, discover the best hidden state sequence S.
- **Problem 2 (Computing Likelihood)**: Given an HMM $\lambda = (A, B)$ and an observation sequence O, determine the likelihood $P(O|\lambda)$.
- **Problem 3 (Learning)** : Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.

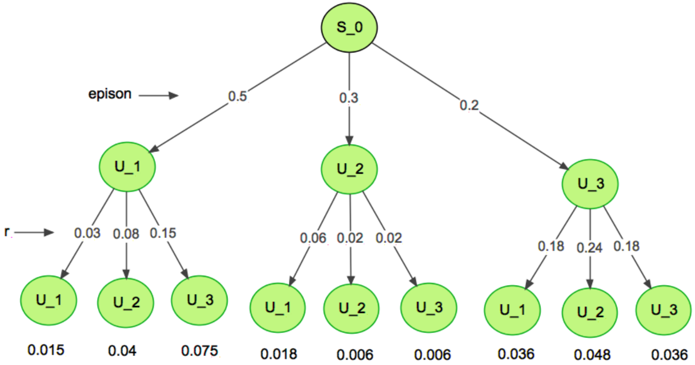
- **Problem 1 (Decoding)**: Given an observation sequence O and an HMM $\lambda = (A, B)$, discover the best hidden state sequence S.

Why is it difficult?

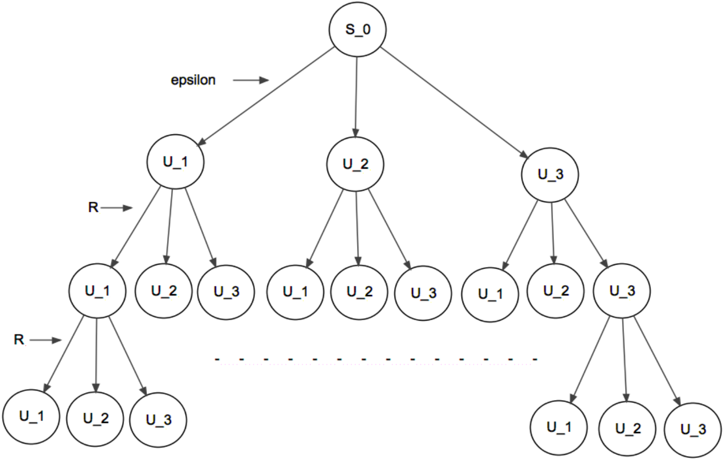


Even if there were only four POS tags, then this is just one of 4x4x4x4 = 256 possible state sequences!

Viterbi Algorithm for the Urn problem (first two symbols)



HMM - Computational Complexity

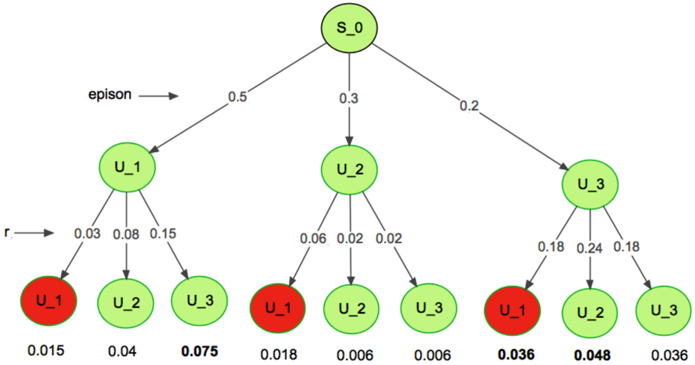


HMM - Computational Complexity

- if the tree is grown in this manner
  - RRGGBRGR - Observation Sequence length = 9 (including epsilon)
  - at each level multiply the node by 3
  - level 1 (ϵ) - 3^1, at level 2 (R) - 3^2, ...at level 9 (R) - 3^9 (nodes at leaf)
  - complexity without restriction = $|S|^{|o|}$

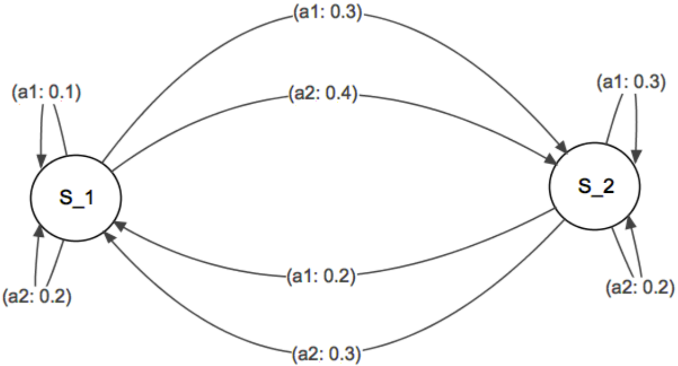
    $|S|$ = Number o States, $|O|$ = length of the observation sequence

Viterbi Algorithm for the Urn problem (first two symbols)

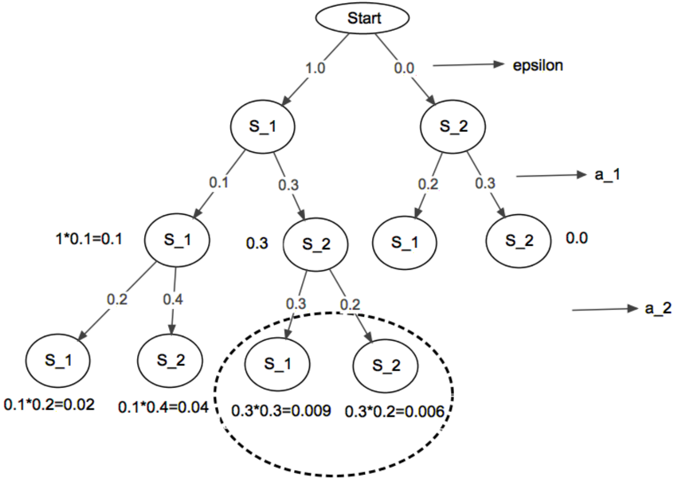


- At every stage, we only keep three nodes
- at the end of observation sequence - we have three nodes (total nodes - 3 x 8)
- complexity comes down from $|S|^{|o|}$ to $|S|.|o|$

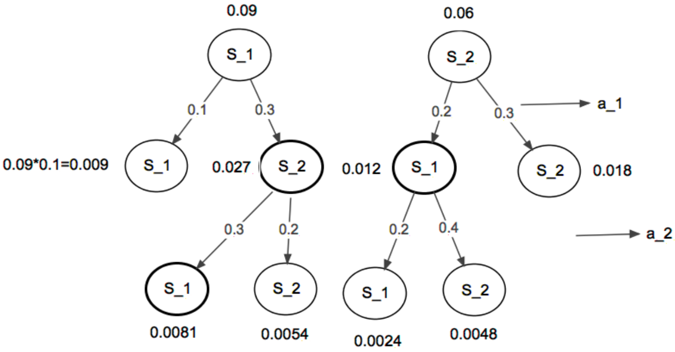
Probabilistic FSM



Probabilistic FSM (contd.)



Probabilistic FSM (contd.)



Tabular Representation of the Tree

	ϵ	a_1	a_2	a_1	a_2
S_1	1.0	(1.0*0.1,0.0*0.2) = (**0.1**,0.0)	(0.02, **0.09**)	(0.009, **0.012**)	(0.0024, **0.0081**)
S_2	0.0	(1.0*0.3,0.0*0.3) = (**0.3**,0.0)	(0.04, **0.06**)	(**0.027**, 0.018)	(0.0048, 0.0054)

- Number of columns = length of observation sequence +1 (ϵ)
- Rows - ending state

HMM - POS Tagging

Goal: choose the most probable tag sequence given the observation sequence of n words $\hat{w}_1^n$

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

Using Bayes' rule

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

Simplifying further by dropping the denominator

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

HMM - POS Tagging

HMM makes two further assumptions:

1. probability of a word depends only on its tag and is independent of neighbouring words and tags

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

2. probability of a word depends only on its tag and is independent of neighbouring words and tags

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

Using these simplifications:

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n|w_1^n) \approx \underset{t_1^n}{\operatorname{argmax}} \prod_{i=1}^{n} \overbrace{P(w_i|t_i)}^{\text{emission}} \overbrace{P(t_i|t_{i-1})}^{\text{transition}}$$

HMM - POS Tagging

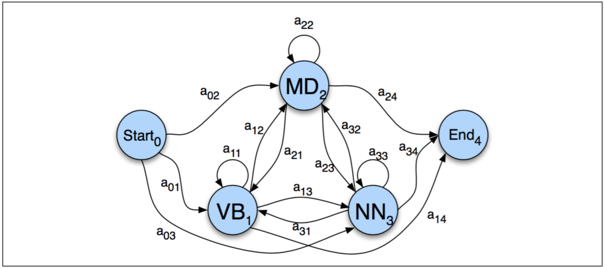


Figure: Markov chain corresponding to the hidden states of HMM. The transition probabilities A are used to compute the prior probability.

HMM - POS Tagging

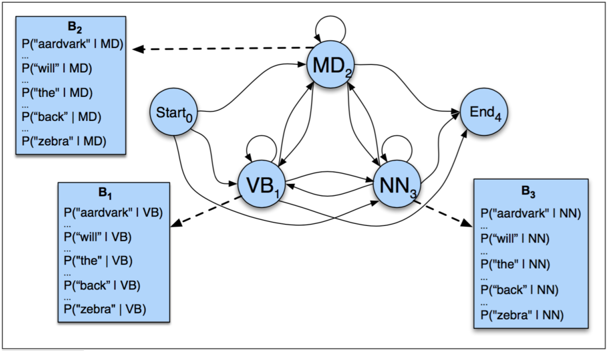


Figure: Observation likelihoods *B* for the HMM.

HMM - POS Tagging

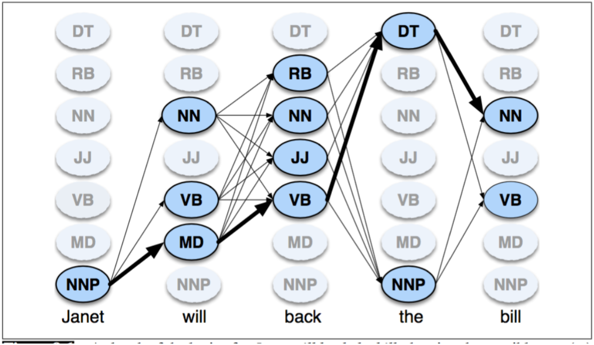


Figure: Observation likelihoods B for the HMM.

Viterbi Algorithm - Pseudocode

function VITERBI(*observations* of len *T*,*state-graph*) **returns** *best-path*

num-states ← NUM-OF-STATES(*state-graph*)
Create a path probability matrix *viterbi[num-states+2,T+2]*
viterbi[0,0] ← 1.0
for each time step t **from** 1 **to** T **do**
  for each state s **from** 1 **to** *num-states* **do**
    $viterbi[s,t] \leftarrow \max_{1 \leq s' \leq num\text{-}states} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$
    $backpointer[s,t] \leftarrow \operatorname{argmax}_{1 \leq s' \leq num\text{-}states} viterbi[s',t-1] * a_{s',s}$
Backtrace from highest probability state in final column of *viterbi[]* and return path

Figure 6.10 Viterbi algorithm for finding optimal sequence of tags. Given an observation sequence and an HMM $\lambda = (A,B)$, the algorithm returns the state-path through the HMM which assigns maximum likelihood to the observation sequence. Note that states 0 and N+1 are non-emitting *start* and *end* states.

POS Tagging - Example

- Janet will back the bill
- Janet/NNP will/MD back/VB the/DT bill/NN

	NNP	MD	VB	JJ	NN	RB	DT
$<s>$	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

POS Tagging - Example

- Janet will back the bill
- Janet/NNP will/MD back/VB the/DT bill/NN

	Janet	**will**	**back**	**the**	**bill**
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0.000097	0
NN	0	0.000200	0.000223	0.000006	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0

POS Tagging - Example

- Janet will back the bill
- Janet/NNP will/MD back/VB the/DT bill/NN