CS626: Speech, NLP and the Web

Penn TAG Set, HMM and Viterbi Decoding, Other Graphical Models for NLP, SVM

Pushpak Bhattacharyya

Computer Science and Engineering Department

IIT Bombay

Week of 24th August, 2020

Task *vs.* Technique Matrix

Task (row) vs. Technique (col) Matrix	**Rules Based/Knowledge-Based**	**Classical ML**				**Deep Learning**		
		Perceptron	Logistic Regression	SVM	Graphical Models (HMM, MEMM, CRF)	Dense FF with BP and softmax	RNN-LSTM	CNN
Morphology								
POS								
Chunking								
Parsing								
NER, MWE								
Coref								
WSD								
Machine Translation								
Semantic Role Labeling								
Sentiment								
Question Answering								

Part of Speech Tagging

- Attach to each word a tag from **Tag-Set**

- Standard Tag-set : Penn Treebank (for English).

Penn POS TAG Set

1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential *there*
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjun
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass
13.	NNS	Noun, plural
14.	NNP	Proper noun, singular
15.	NNPS	Proper noun, plural
16.	PDT	Predeterminer
17.	POS	Possessive ending
18.	PRP	Personal pronoun
19.	PRP$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative

Penn POS TAG Set (cntd)

22.	RBS	Adverb, superlative
23.	RP	Particle
24.	SYM	Symbol
25.	TO	*to*
26.	UH	Interjection
27.	VB	Verb, base form
28.	VBD	Verb, past tense
29.	VBG	Verb, gerund or present participle
30.	VBN	Verb, past participle
31.	VBP	Verb, non-3rd person singular present
32.	VBZ	Verb, 3rd person singular present
33.	WDT	Wh-determiner
34.	WP	Wh-pronoun
35.	WP$	Possessive wh-pronoun
36.	WRB	Wh-adverb

A dialogue text POS tagged from Treebank

[SpeakerB1/SYM]

./.

So/UH how/WRB many/JJ ,/, um/UH ,/,

[credit/NN cards/NNS]

do/VBP

[you/PRP]

have/VB ?/.

[SpeakerA2/SYM]

./.

[Um/UH]

,/,

[I/PRP]

think/VBP

[I/PRP]

'm/VBP down/IN to/IN

[one/CD]

./.

https://catalog.ldc.upenn.edu/desc/addenda/LDC99T42.pos.txt

Mathematics of POS tagging

Noisy Channel Model

W *Noisy Channel* **T**

$(w_n, w_{n-1}, \ldots, w_1)$ $(t_m, t_{m-1}, \ldots, t_1)$

Sequence *W* is transformed into sequence *T*

$$T^*=\underset{T}{\operatorname{argmax}}(P(T|W))$$

$$W^*=\underset{W}{\operatorname{argmax}}(P(W|T))$$

Argmax computation (1/2)

Best tag sequence
= T*
= argmax P(T|W)
= argmax P(T)P(W|T) (by Baye's Theorem)

$$\mathbf{P(T) = P(t_0{=}\text{\textasciicircum}\ t_1 t_2 \ldots t_{n+1}{=}.)}$$

$$\mathbf{= P(t_0)P(t_1|t_0)P(t_2|t_1t_0)P(t_3|t_2t_1t_0) \ldots P(t_n|t_{n-1}t_{n-2}\ldots t_0)P(t_{n+1}|t_nt_{n-1}\ldots t_0)}$$

$$\mathbf{= P(t_0)P(t_1|t_0)P(t_2|t_1) \ldots P(t_n|t_{n-1})P(t_{n+1}|t_n)}$$

$$= \prod_{i=0}^{N+1} \mathbf{P(t_i|t_{i-1})}$$ Bigram Assumption

Argmax computation (2/2)

$$P(W|T) = P(w_0|t_0\text{-}t_{n+1})P(w_1|w_0t_0\text{-}t_{n+1})P(w_2|w_1w_0t_0\text{-}t_{n+1}) \ldots P(w_n|w_0\text{-}w_{n-1}t_0\text{-}t_{n+1})P(w_{n+1}|w_0\text{-}w_nt_0\text{-}t_{n+1})$$

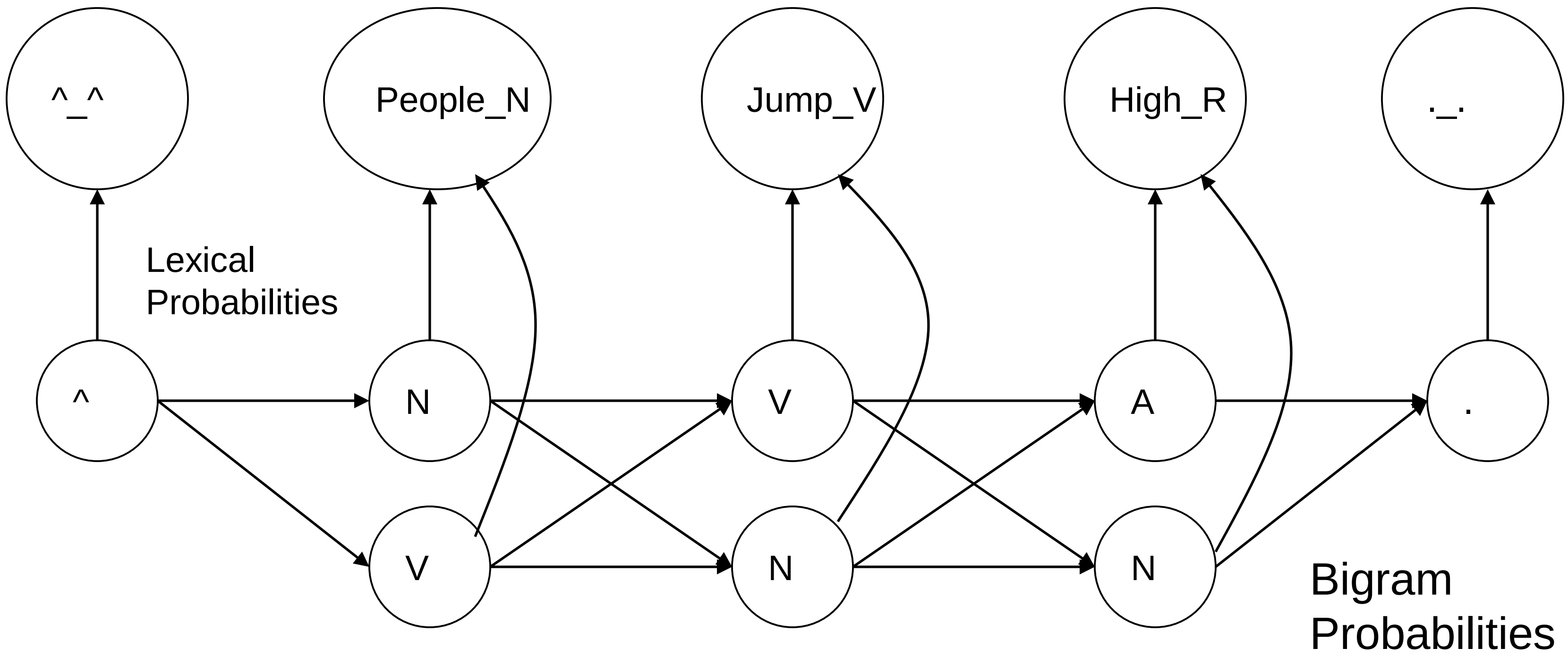
Assumption: A word is determined completely by its tag. This is inspired by speech recognition

$$= P(w_o|t_o)P(w_1|t_1) \ldots P(w_{n+1}|t_{n+1})$$

$$= \prod_{i=0}^{n+1} P(w_i|t_i)$$

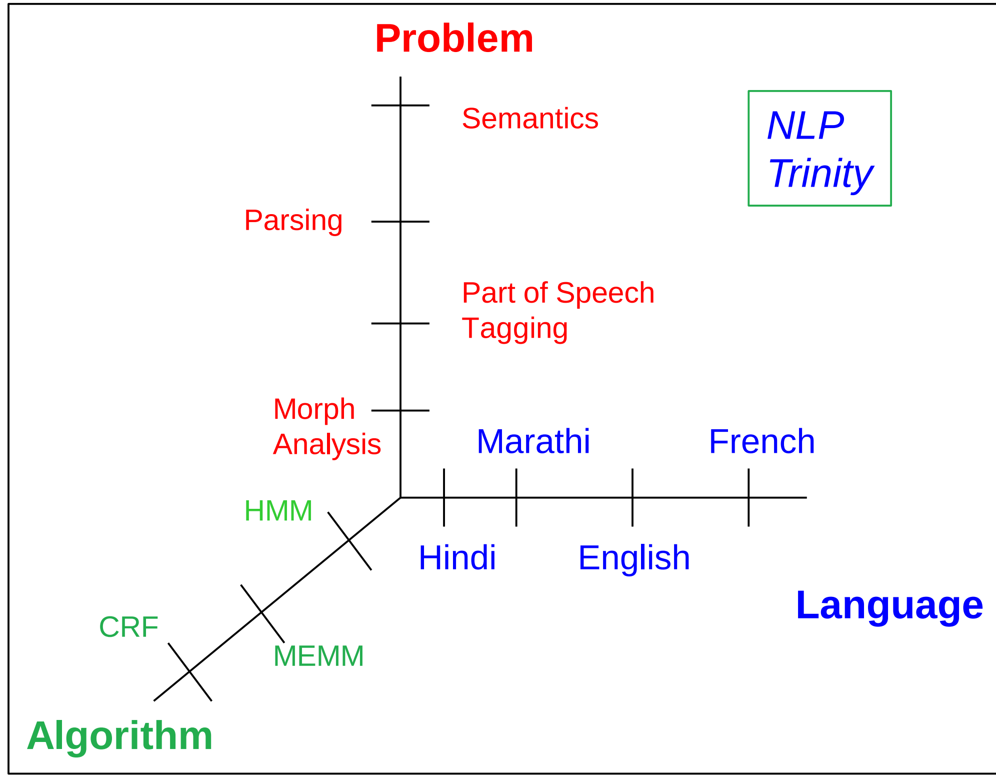
$$= \prod_{i=1}^{n+1} P(w_i|t_i) \quad \text{(Lexical Probability Assumption)}$$

Generative Model



This model is called Generative model.
Here words are observed from tags as states.
This is similar to HMM.

HMM



Lawrence R. Rabiner:a tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 1989, pages 257–286

https://web.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/tutorial%20on%20hmm%20and%20applications.pdf

Definition of HMM and URN example

- An HMM is defined by

  <S, V, A, B, π>

Here :

- $S = \{U1, U2, U3\}$

- $V = \{ R,G,B\}$

For observation sequence:

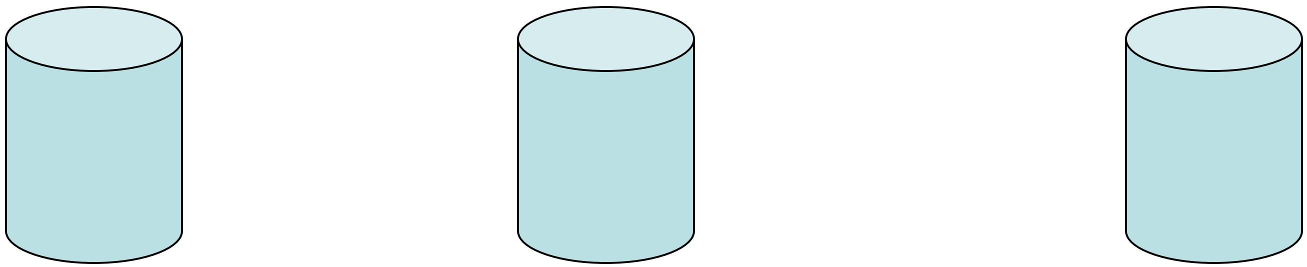
$$O = [O_1 \dots O_n]$$

and State sequence

$$Q = [S_1 \dots S_n]$$

URN Example

Colored Ball choosing



Urn 1	Urn 2	Urn 3
# of Red = 30	# of Red = 10	# of Red =60
# of Green = 50	# of Green = 40	# of Green =10
# of Blue = 20	# of Blue = 50	# of Blue = 30

Viterbi Decoding to find state sequence

- Observation : RRGGBRGR

- Find best possible state sequence

Noting probabilities again

A =

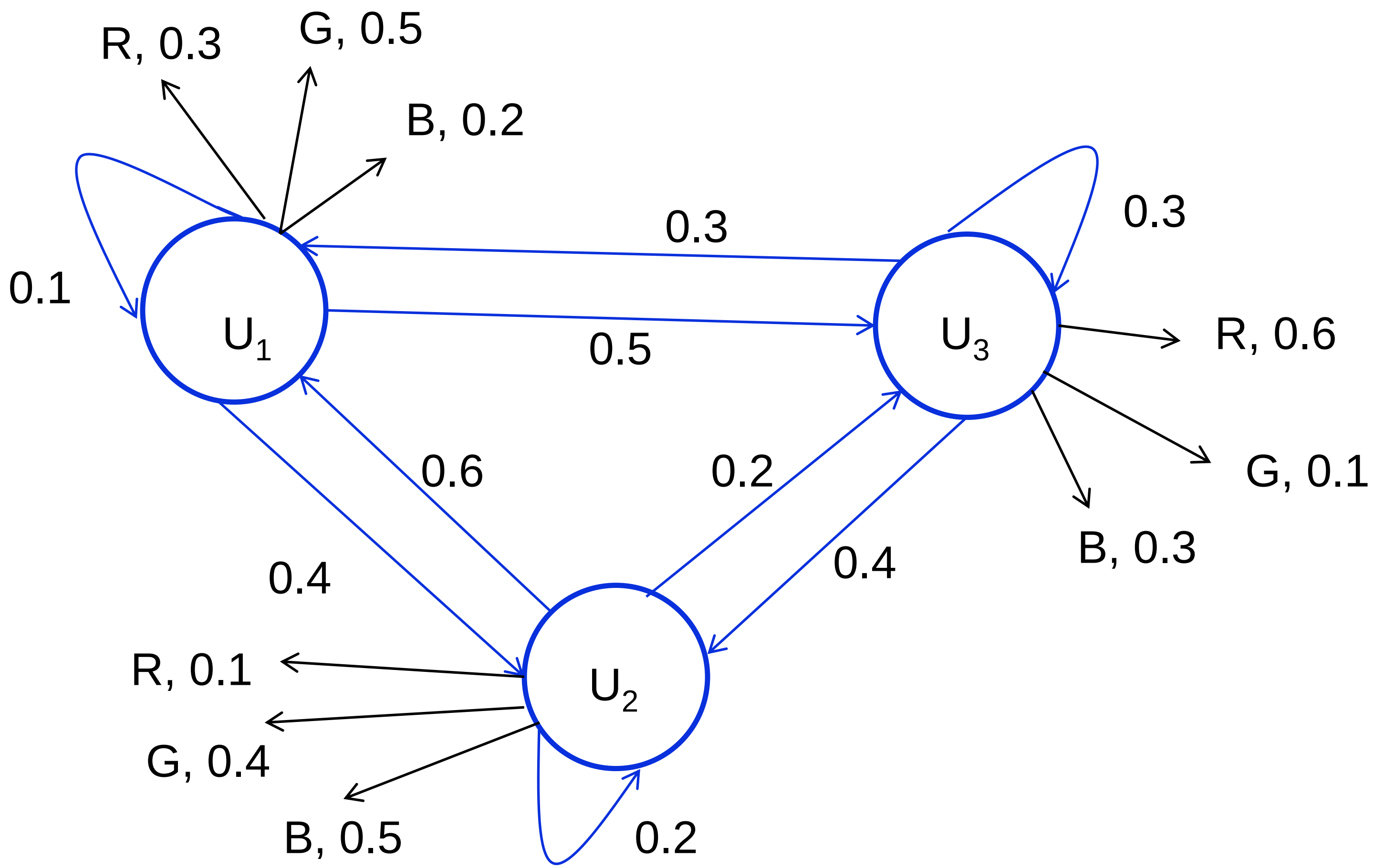
	U_1	U_2	U_3
U_1	0.1	0.4	0.5
U_2	0.6	0.2	0.2
U_3	0.3	0.4	0.3

B=

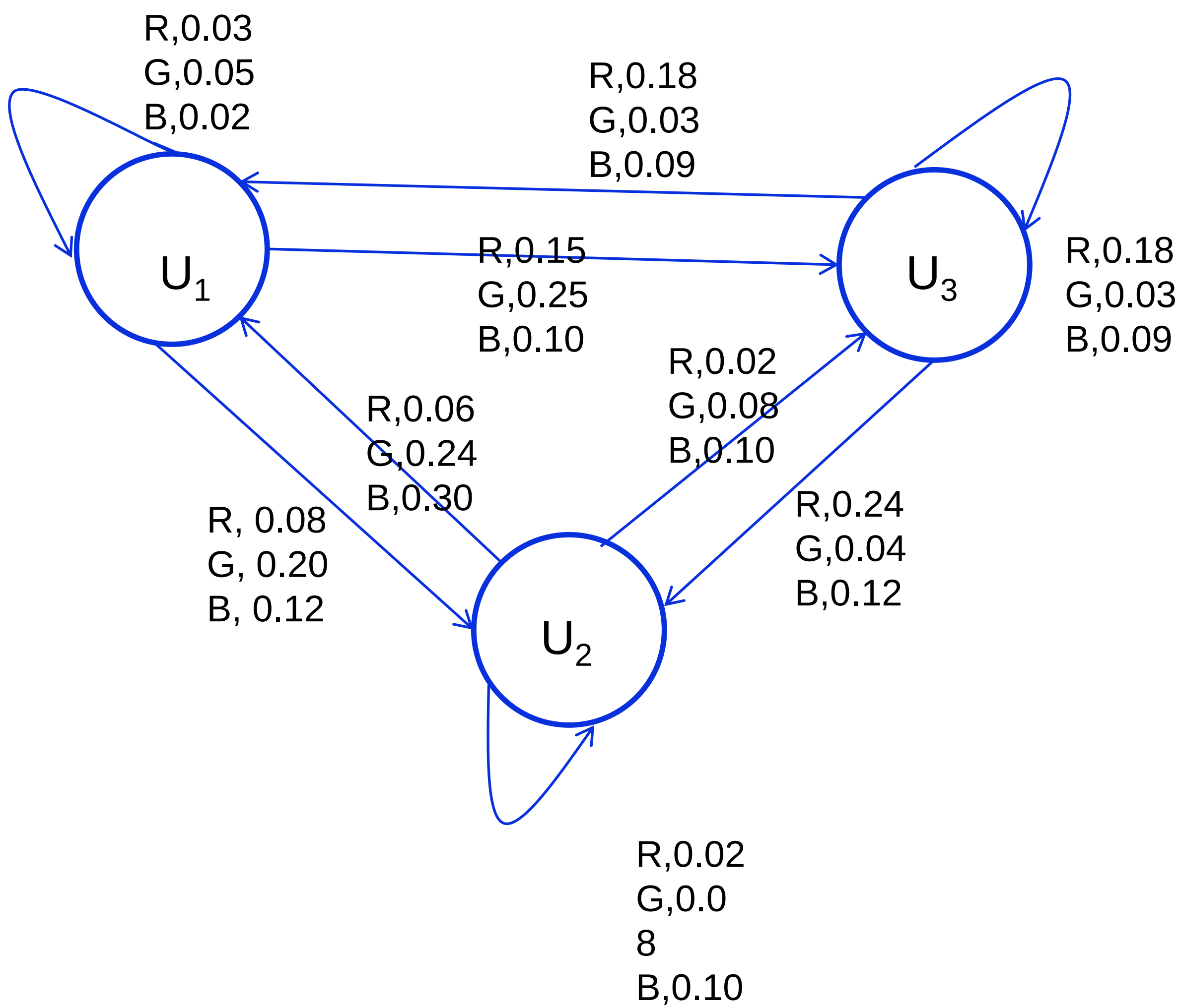
	R	G	B
U_1	0.3	0.5	0.2
U_2	0.1	0.4	0.5
U_3	0.6	0.1	0.3

$$\pi_i = P(q_1 = U_i)$$

Diagrammatic representation (1/2)



Diagrammatic representation (2/2)



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* = “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

$$S^* = \arg\max_S (P(S \mid O))$$

False Start

	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

$$P(S\,|\,O) = P(S_{1-8}\,|\,O_{1-8})$$

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

By Markov Assumption (a state depends only on the previous state)

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

Bayes Theorem

$$P(A \mid B) = P(A).P(B \mid A) / P(B)$$

P(A) -: Prior

P(B|A) -: Likelihood

$$\arg\max_S P(S \mid O) = \arg\max_S P(S).P(O \mid S)$$

State Transitions Probability

$$P(S) = P(S_{1-8})$$

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

By Markov Assumption (k=1)

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

Observation Sequence probability

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

Assumption that ball drawn depends only on the Urn chosen

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

$$P(S \mid O) = P(S).P(O \mid S)$$

$$P(S \mid O) = P(S_1).P(S_2 \mid S_1).P(S_3 \mid S_2).P(S_4 \mid S_3)...P(S_8 \mid S_7).$$

$$P(O_1 \mid S_1).P(O_2 \mid S_2).P(O_3 \mid S_3)...P(O_8 \mid S_8)$$

Grouping terms

	O_0	O_1	O_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

$P(S).P(O|S)$

$= [P(O_0|S_0).P(S_1|S_0)].$

$[P(O_1|S_1). \quad P(S_2|S_1)].$

$[P(O_2|S_2). \quad P(S_3|S_2)].$

$[P(O_3|S_3).P(S_4|S_3)].$

$[P(O_4|S_4).P(S_5|S_4)].$

$[P(O_5|S_5).P(S_6|S_5)].$

$[P(O_6|S_6).P(S_7|S_6)].$

$[P(O_7|S_7).P(S_8|S_7)].$

$[P(O_8|S_8).P(S_9|S_8)].$

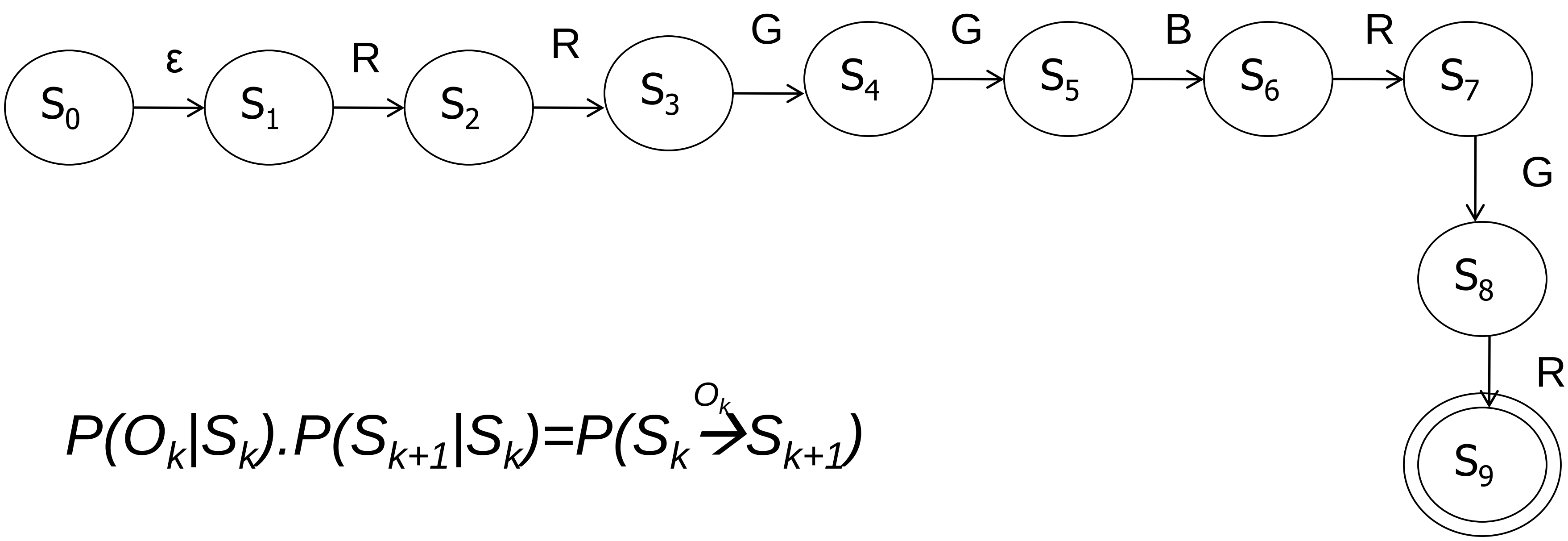
We introduce the states S_0 and S_9 as initial and final states respectively.

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

Introducing useful notation

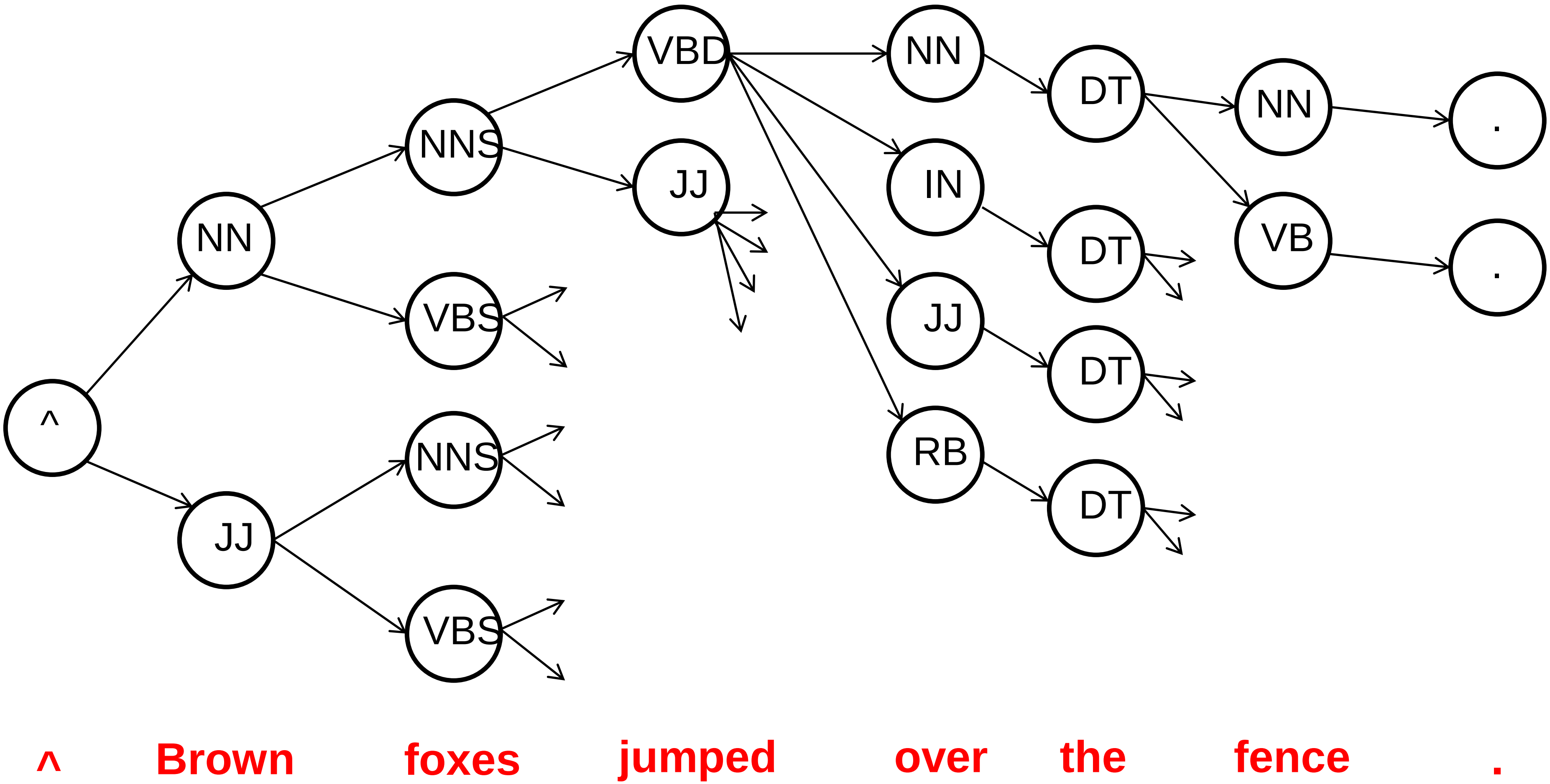
	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	
Obs:	ε	R	R	G	G	B	R	G	R	
State:	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9

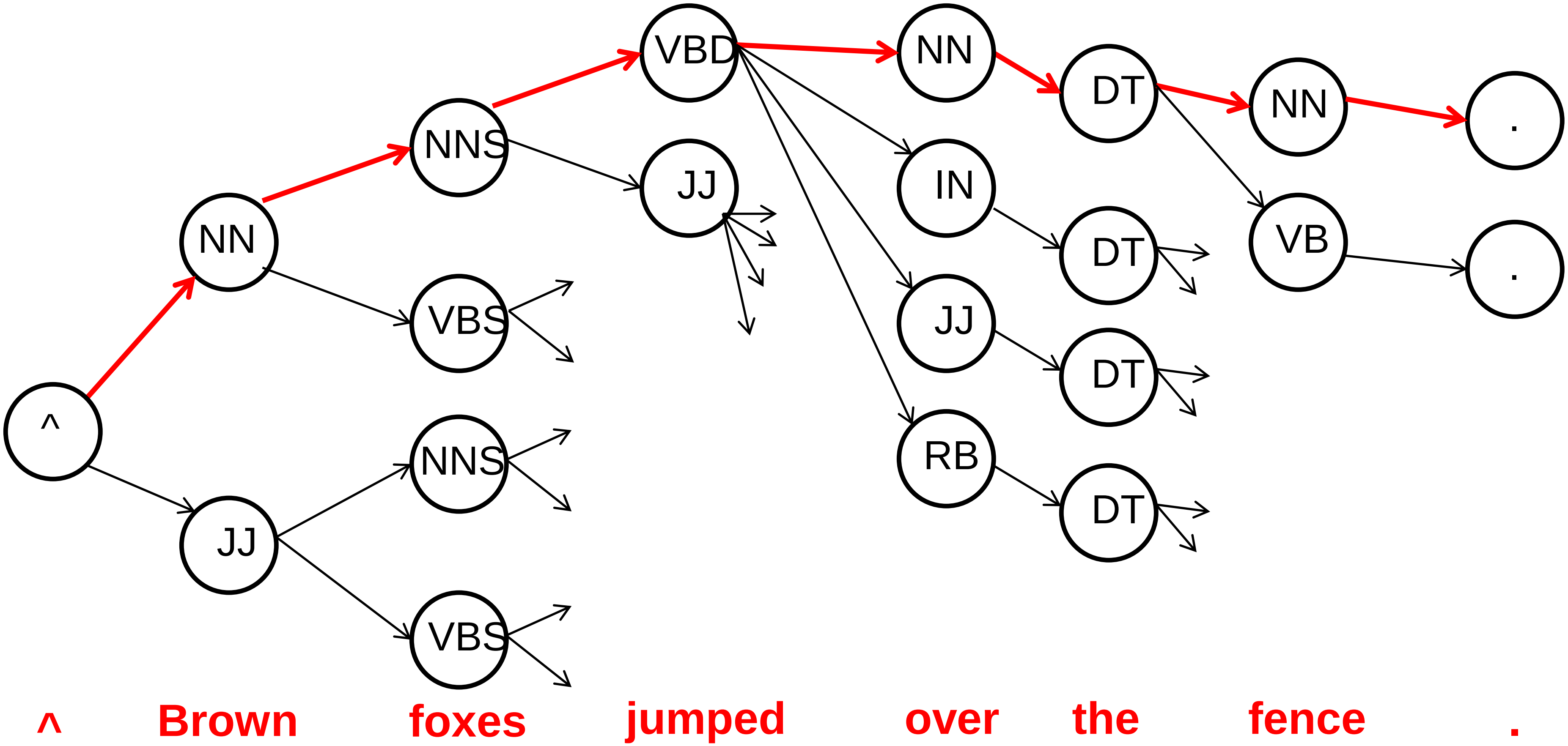


$$P(O_k|S_k).P(S_{k+1}|S_k)=P(S_k \xrightarrow{O_k} S_{k+1})$$

Recall

W:	^	Brown	foxes	jumped	over	the	fence	.
T:	^	JJ	NNS	VBD	NN	DT	NN	.
		NN	VBS	JJ	IN		VB	
					JJ			
					RB			





Probability of a path (e.g. Top most path) = *P*(T) * *P*(W|T)

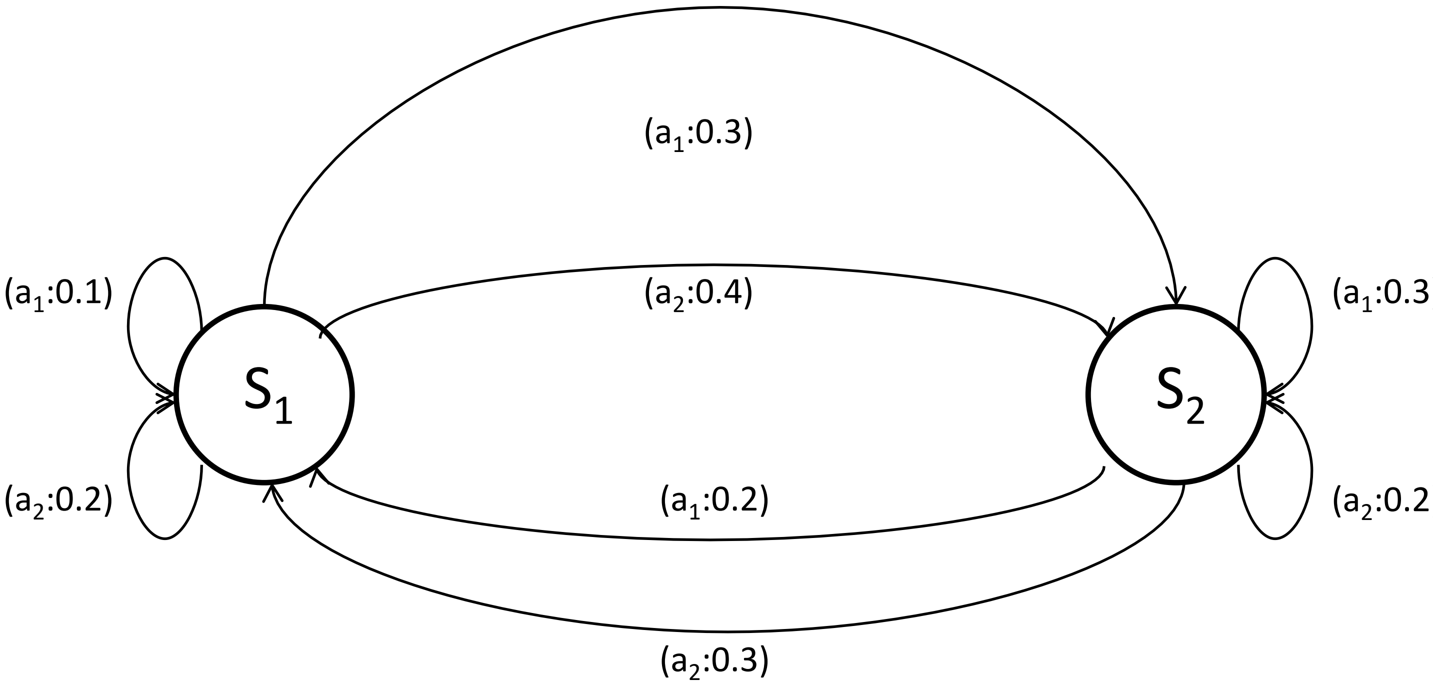
P(^) . P(NN|^) . P(NNS|NN) . P(VBD|NNS) . P(NN|VBD) . P(DT|NN) . P(NN|DT) . P(.|NN) . P(.)

*

P(^|^) . P(brown|NN) . P(foxes|NNS) . P(jumped|VBD) . P(over|NN) . P(the|DT) . P(fence|NN) . P(.|.)

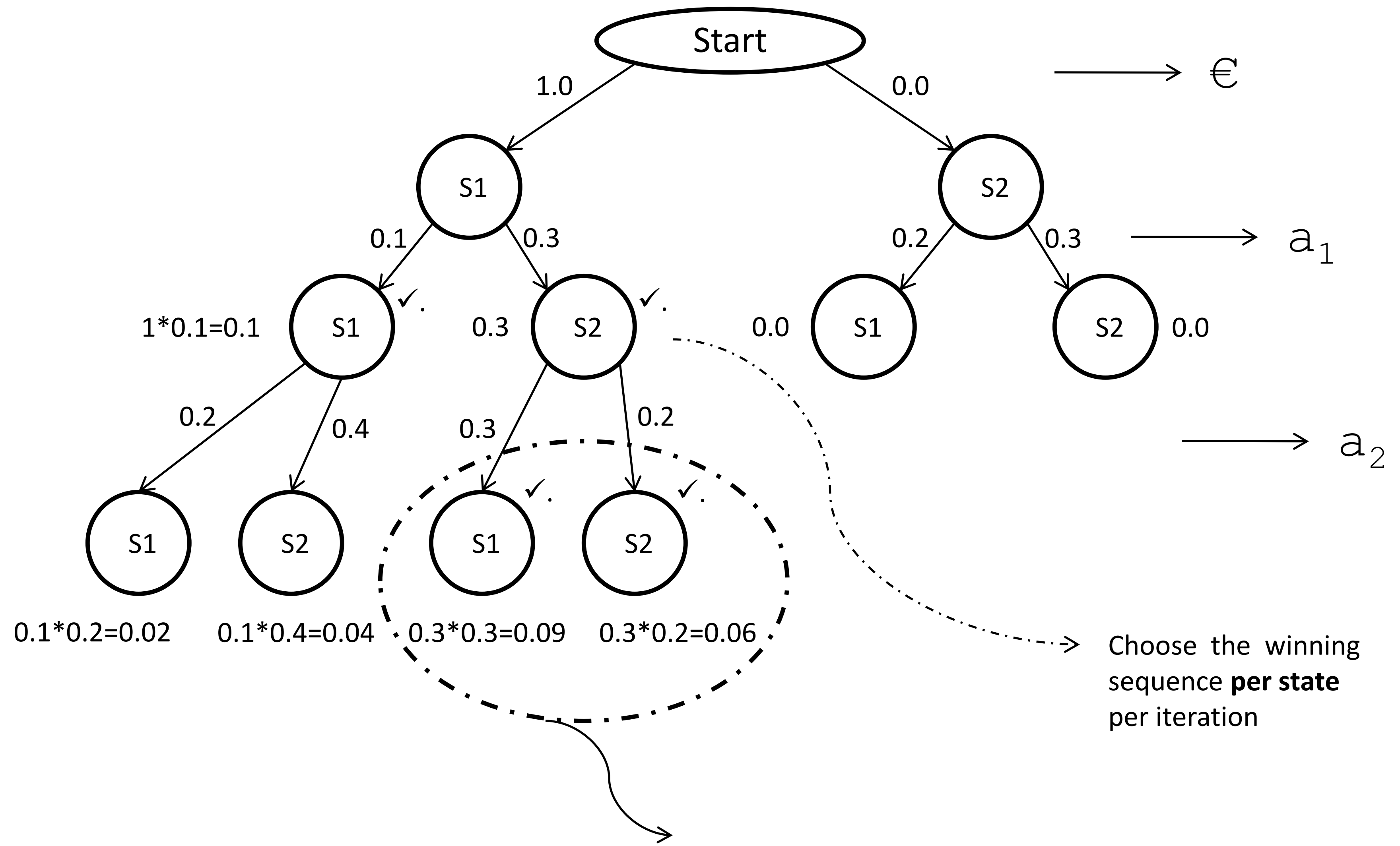
Viterbi Decoding

Probabilistic FSM

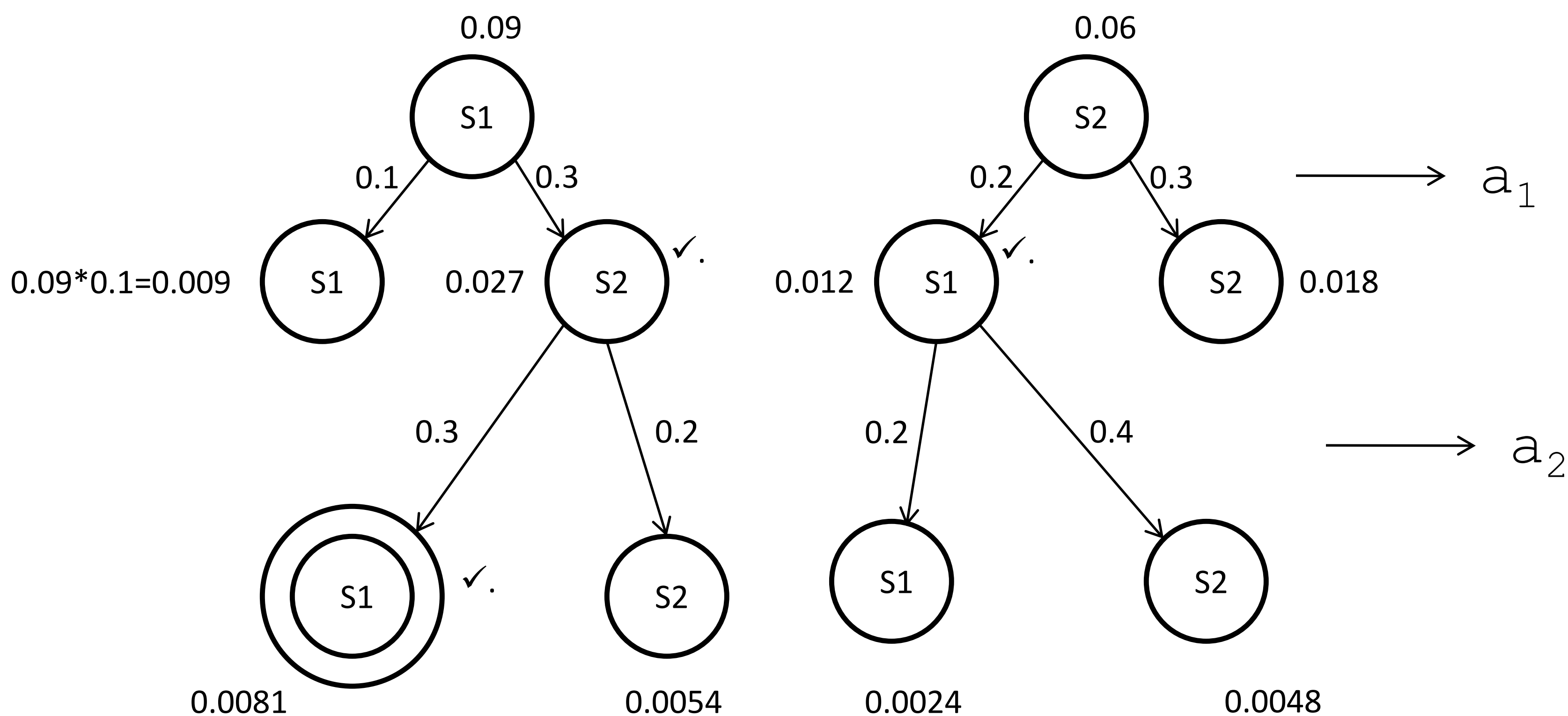


The question here is:
“what is the most likely state sequence given the output sequence seen”

Developing the tree



Tree structure contd…



The problem being addressed by this tree is $S^* = \arg\max_{s} P(S \mid a_1 - a_2 - a_1 - a_2, \mu)$

a1-a2-a1-a2 is the output sequence and μ the model or the machine

Path found:
(working backward)

$$S_1 \xrightarrow{a_1} S_2 \xrightarrow{a_2} S_1 \xrightarrow{a_1} S_2 \xrightarrow{a_2} S_1$$

Problem statement: Find the best possible sequence

$$S^* = \arg\max_{s} P(S \mid O, \mu)$$

$$where,\ S \rightarrow \text{State Seq},\ O \rightarrow \text{Output Seq},\ \mu \rightarrow \text{Model or Machine}$$

$$\text{Model or Machine} = \{S_0, S, A, T\}$$

Start symbol   State collection   Alphabet set   Transitions

T is defined as $P(S_i \xrightarrow{a_k} S_j) \quad \forall i, j, k$

Tabular representation of the tree

Latest symbol observed / Ending state	€	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)

Note: Every cell records the winning probability ending in that state

Final winner

The bold faced values in each cell shows the sequence probability ending in that state. Going backward from final winner sequence which ends in state S_2 (indicated By the 2^{nd} tuple), we recover the sequence.

Algorithm

(following James Alan, Natural Language Understanding (2nd edition), Benjamin Cummins (pub.), 1995

Given:

1. The HMM, which means:
   a. Start State: S_1
   b. Alphabet: $A = \{a_1, a_2, \ldots a_p\}$
   c. Set of States: $S = \{S_1, S_2, \ldots S_n\}$
   d. Transition probability $P(S_i \xrightarrow{a_k} S_j) \quad \forall_{i,\, j,\, k}$
      which is equal to $P(S_j, a_k \mid S_i)$
2. The output string $o_1 o_2 \ldots o_T$

To find:

The most likely sequence of states $C_1 C_2 \ldots C_T$ which produces the given output sequence, *i.e.*, $S_1 S_2 \ldots S_T$ = $\arg\max_C [P(S \mid o_1, o_2, \ldots o_T, \mu)]$

1. Initialization

SEQSCORE(1,1)=1.0

BACKPTR(1,1)=0

For(i=2 to N) do

SEQSCORE(i,1)=0.0

[expressing the fact that first state is S_1]

2. Iteration

For(t=2 to T) do

For(i=1 to N) do

SEQSCORE(i,t) = $\text{Max}_{(j=1,N)}$

$$[SEQSCORE(j,(t-1)) * P(Sj \xrightarrow{a_k} Si)]$$

BACKPTR(I,t) = index *j* that gives the MAX above

3. Seq. Identification

S(T) = i that maximizes SEQSCORE(i,T)

  For i from (T-1) to 1 do

    S(i) = BACKPTR[S(i+1),(i+1)]

Optimizations possible:

1. BACKPTR can be 1*T
2. SEQSCORE can be T*2

Homework:- Compare this with A*, Beam Search [Homework]

Reason for this comparison:

Both of them work for finding and recovering sequence

Back to POS tag problem

Viterbi for POS Tagging

- E.g.
  - T: Tags
  - W: Words
  - Two special symbol: ‘^’ and ‘.’

  Find out number of paths in the tree given word sequence.

Exponential *w.r.t.* number of words in the sentence of length *L*

Number of path = Number of leaves in the tree.

$$O(T^L)$$

How to avoid it?

We do not need exponential work!

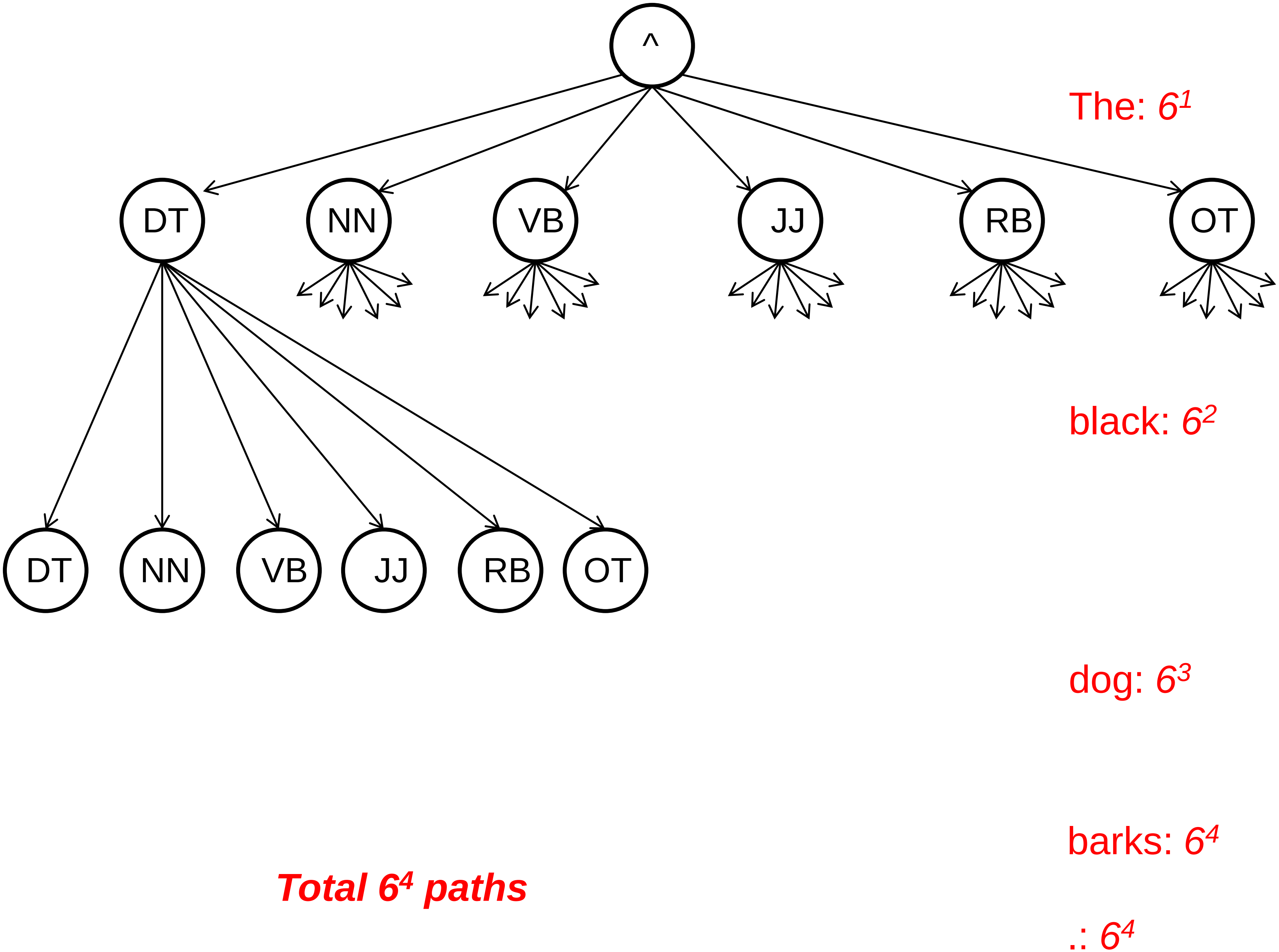
- Suppose our tags are
  - DT, NN, VB, JJ, RB and OT
- E.g.

DT- determiner
NN- Noun
VB- Verb
JJ- Adjective
RB- Adverb
OT- others

^	The	black	dog	barks	.
^	DT	DT	DT	DT	.
	NN	NN	NN	NN	
	VB	VB	VB	VB	
	JJ	JJ	JJ	JJ	
	RB	RB	RB	RB	
	OT	OT	OT	OT	

Possible tags

So, 6^4 possible path



^
DT
NN
VB
JJ
RB
OT
DT
NN
VB
JJ
RB
OT
The: 6^1
black: 6^2
dog: 6^3
barks: 6^4
.: 6^4
Total 6^4 paths

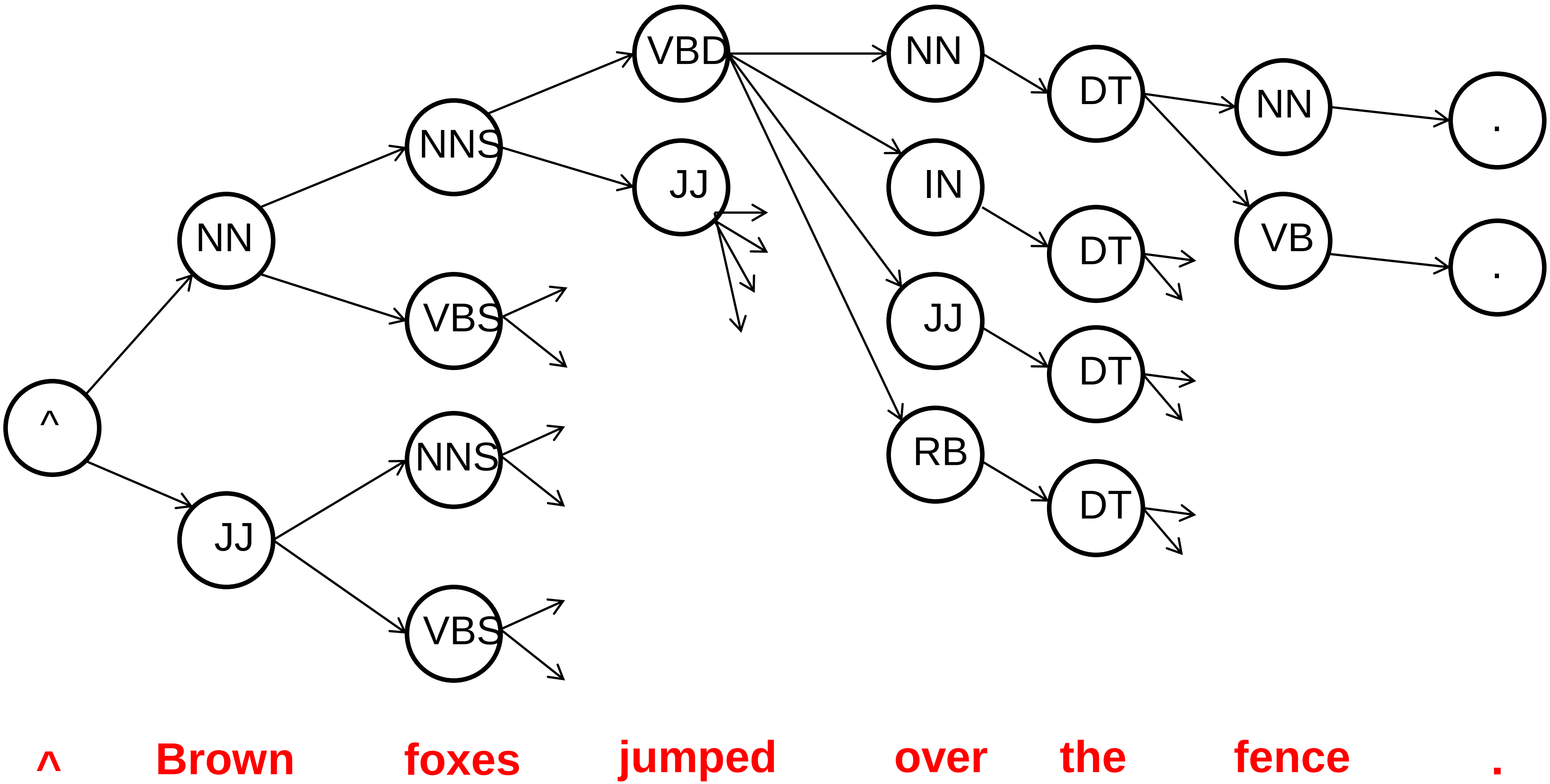
Consider the paths that end in NN after seeing input “The black”

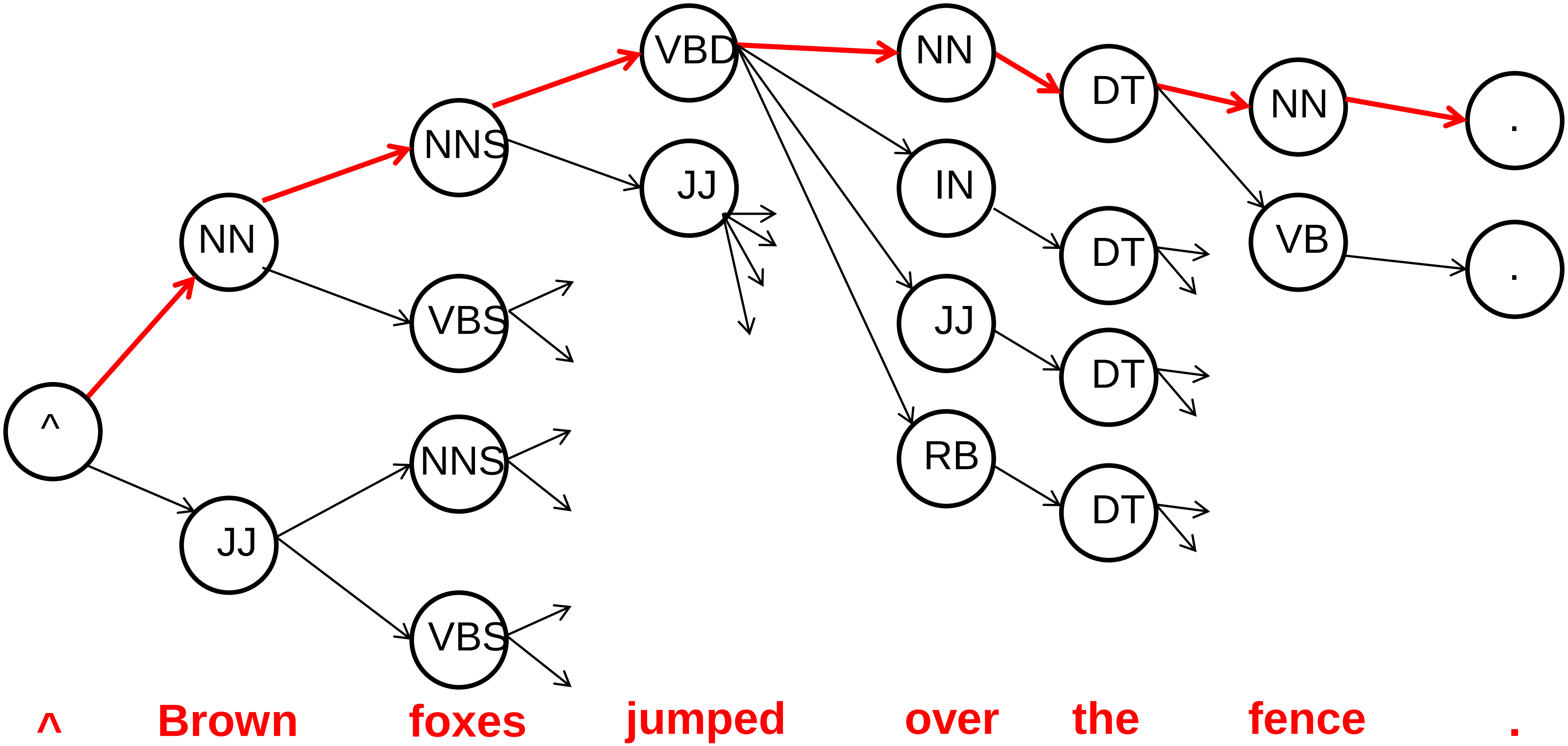
^	**The**	**black**	
^	DT	NN	***P*(T).*P*(W|T) =** *P*(DT|^) . *P*(NN|DT) . *P*(The|DT) . *P*(Black|NN)
^	NN	NN	***P*(T).*P*(W|T) =** *P*(NN|^) . *P*(NN|NN) . *P*(The|NN) . *P*(Black|NN)
^	VB	NN	***P*(T).*P*(W|T) =** *P*(VB|^) . *P*(NN|VB) . *P*(The|VB) . *P*(Black|NN)
^	JJ	NN	***P*(T).*P*(W|T) =** *P*(JJ|^) . *P*(NN|JJ) . *P*(The|JJ) . *P*(Black|NN)
^	RB	NN	***P*(T).*P*(W|T) =** *P*(RB|^) . *P*(NN|RB) . *P*(The|RB) . *P*(Black|NN)
^	OT	NN	***P*(T).*P*(W|T) =** *P*(OT|^) . *P*(NN|OT) . *P*(The|OT) . *P*(Black|NN)

$$Complexity = L * T^2$$

For each tag, only path with ***highest probability*** value are ***retained***, others are ***discarded***.

W:	^	Brown	foxes	jumped	over	the	fence	.
T:	^	JJ	NNS	VBD	NN	DT	NN	.
		NN	VBS	JJ	IN		VB	
					JJ			
					RB			





Probability of a path (e.g. Top most path) = *P*(T) * *P*(W|T)

$P(\hat{}) \,.\, P(NN|\hat{}) \,.\, P(NNS|NN) \,.\, P(VBD|NNS) \,.\, P(NN|VBD) \,.\, P(DT|NN) \,.\, P(NN|DT) \,.\, P(.|NN) \,.\, P(.)$

*

$P(\hat{}|\hat{}) \,.\, P(brown|NN) \,.\, P(foxes|NNS) \,.\, P(jumped|VBD) \,.\, P(over|NN) \,.\, P(the|DT) \,.\, P(fence|NN) \,.\, P(.|.)$

Decoding Summary

- On every word compute the partial path probability
- Out of all partial paths ending in a particular state, choose the one with highest path probability
- Advance only that leaf
- In case of tie, choose any one arbitrarily

Assignment Discussion

Brown Corpus

- 1,014,312 words of running text of edited English prose printed in the United States
- 500 samples of 2000+ words each
- Facilitate automatic or semi-automatic syntactic analysis

Universal POS Tag Set (https://universaldependencies.org/u/pos/)

Open class words	Closed class words	Other
ADJ (*The car is **green**.)*	ADP	PUNCT
ADV (***arguably** wrong)*	AUX	SYM
INTJ (*yes*, *no*, *uhuh*, etc.)	CCONJ	X
NOUN (tree, man)	DET	
PROPN	NUM	
VERB	PART	
	PRON	
	SCONJ	

Noun (1/2)

Definition

- Nouns are a part of speech typically denoting a person, place, thing, animal or idea.
- The NOUN tag is intended for common nouns only. See PROPN for proper nouns and PRON for pronouns.
- Note that some verb forms such as *gerunds* and *infinitives* may share properties and usage of nouns and verbs. Depending on language and context, they may be classified as either VERB or NOUN.

Swimming_noun is a good exercise; He is swimming_verb

Noun (2/2)

Examples

- *girl*
- *cat*
- *tree*
- *air*
- *beauty*

References

- Loos, Eugene E., et al. 2003. Glossary of linguistic terms: What is a noun?
- Wikipedia

Annotation matter

Tag repository and probability

- Where do tags come from?
  - Tag set
- How to get probability values i.e. P(.)?
  - Annotated corpora

After modeling of the problem, emphasis should be on the corpus

Computing P(.) values

Let us suppose annotated corpus has the following sentence

I	**have**	**a**	**brown**	**bag**	**.**
PRN	**VB**	**DT**	**JJ**	**NN**	**.**

$$P(NN \mid JJ) = \frac{Number\_of\_times\_JJ\_followed\_by\_NN}{Number\_of\_times\_JJ\_appeared}$$

$$P(Brown \mid JJ) = \frac{Number\_of\_times\_Brown\_tagged\_as\_JJ}{Number\_of\_times\_JJ\_appeared}$$

Why Ratios?

- This way of computing parameter probabilities: is this correct?
- What does “correct” mean?
- Is this principled?
- We are using Maximum Likelihood Estimate (**MLE**)
- Assumption: underlying distribution is multinomial

Explanation with coin tossing

- A coin is tossed 100 times, Head appears 40 times
- P(H)= 0.4
- Why?
- Because of maximum likelihood

N tosses, K Heads, parameter *P(H)=p*

- Construct Maximum Likelihood Expression
- Take log likelihood and take derivative
- Equate to 0 and Get *p*

$$L = p^K (1-p)^{N-K}$$

$$\Rightarrow LL = \log(L) = K \log p + (N-K)\log(1-p)$$

$$\Rightarrow \frac{d(LL)}{dp} = \frac{K}{p} - \frac{N-K}{1-p}$$

$$\Rightarrow \frac{d(LL)}{dp} = 0 \quad gives \quad p = \frac{K}{N}$$

Exercise

- Following the process for finding the probability of Head from N tosses of coin yielding K Heads, prove that the transition probabilities can be found from MLE
- Most important: get the likelihood expression
- Use chapter 2 of the book
  - Pushpak Bhattacharyya: Machine translation, CRC Press, Taylor & Francis Group, Boca Raton, USA, 2015, ISBN: 978-1-4398-9718-8

Appendix

Appendages to tags in Penn Tag Set

S = plural

$ = possessive

R = comparative

T = superlative

O = objective case of pronoun

D = past tense

Z = 3rd singular verb

N = past participle

G = present participle or gerund

Machine Translation v/s POS tagging!

- Similarity
  - POS
    - Every word in a sentence has one corresponding tag.
  - MT
    - Every word in a sentence has one (or more) corresponding translated word.
- Difference
  - Order: Order of translated word may change.
  - Fertility: One word corresponds to many. Many to one also possible.

Complexity

- POS and HMM
  - Linear time complexity
- MT and Bean search
  - Exponential time complexity
  - Permutation of words produces exponential searc space

  - However, for related languages, MT is like POS tagging

Properties of related languages

1. Order preserving

2. Fertility ~ 1

3. Morphology preserving

Hindi	*Jaaunga*
Bengali	*Jaabo*
English	*Will go*

Hindi & Bengali ↑

Hindi & English ↓

Properties of related languages

4. Syncretism: Suffix features should be similarly loaded

Hindi	Main *jaaunga*	Hum *jaayenge*
Bengali	Ami *jaabo*	Aamra *jaabo*

Hindi & Bengali ↓

5. Idiomaticity: Literal translation should be high

Hindi	*Aap Kaise Ho?*
Bengali	*Aapni Kemon Achen?*
English	*How do you do?*

Hindi & Bengali ↑

Hindi & English ↓

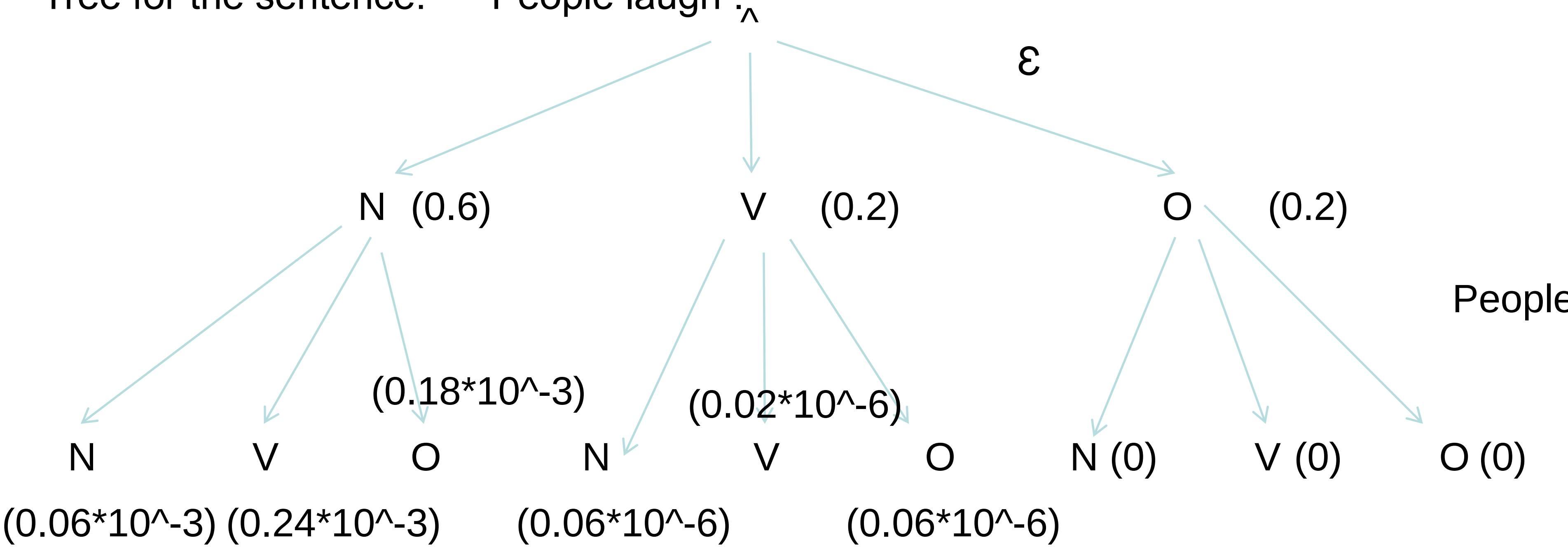
Points to ponder wrt HMM and Viterbi

Viterbi Algorithm

- Start with the start state.
- Keep advancing sequences that are “maximum” amongst all those ending in the same state

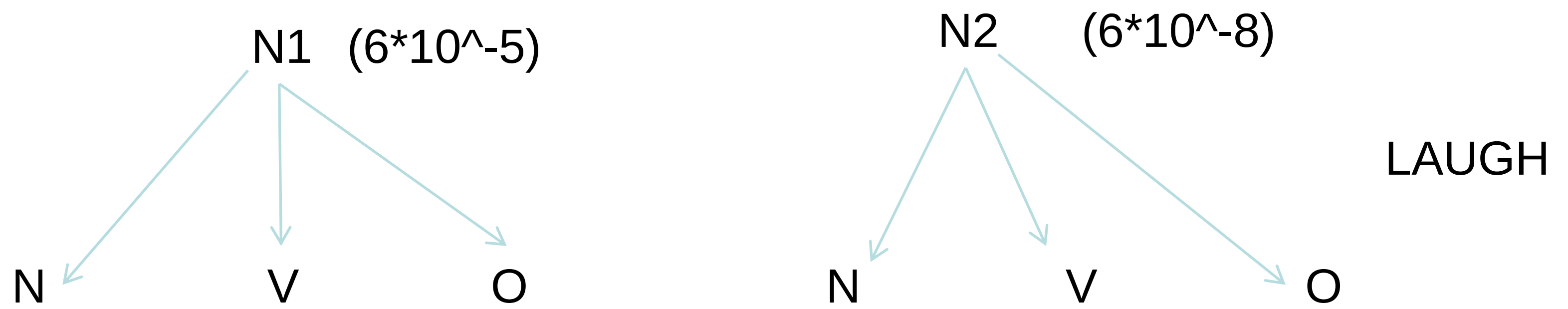
Viterbi Algorithm

Tree for the sentence: “^ People laugh .”



Claim: We do not need to draw all the subtrees in the algorithm

Viterbi phenomenon (Markov process)



Next step all the probabilities will be multiplied by identical probability (lexical and transition). So children of N2 will have probability less than the children of N1.

What does P(A|B) mean?

- P(A|B)= P(B|A)

  If P(A)=P(B)

- P(A|B) means??
  - Causality?? B causes A??
  - Sequentiality?? A follows B?

Classic problems with respect to HMM

1. Given the observation sequence, find the possible state sequences- Viterbi
2. Given the observation sequence, find its probability- forward/backward algorithm
3. Given the observation sequence find the HMM prameters.- Baum-Welch algorithm

Illustration of Viterbi

- The “start” and “end” are important in a sequence.
- Subtrees get eliminated due to the Markov Assumption.

<u>POS Tagset</u>

- N(noun), V(verb), O(other)   [simplified]
- ^ (start), . (end)   [start & end states]

Illustration of Viterbi

<u>Lexicon</u>

people: N, V

laugh: N, V

.

.

.

<u>Corpora for Training</u>

________^ $w_{11}\_t_{11}$ $w_{12}\_t_{12}$ $w_{13}\_t_{13}$ ....$w_{1k\_1}\_t_{1k\_1}$.

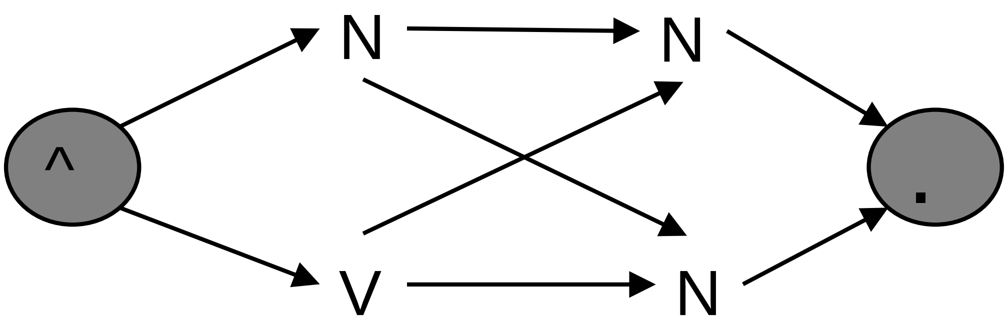
^ $w_{21}\_t_{21}$ $w_{22}\_t_{22}$ $w_{23}\_t_{23}$ ....$w_{2k\_2}\_t_{2k\_2}$.

.

.

^ $w_{n1}\_t_{n1}$ $w_{n2}\_t_{n2}$ $w_{n3}\_t_{n3}$ ....$w_{nk\_n}\_t_{nk\_n}$.

Inference



Partial sequence graph

	^	N	V	O	.
^	0	0.6	0.2	0.2	0
N	0	0.1	0.4	0.3	0.2
V	0	0.3	0.1	0.3	0.3
O	0	0.3	0.2	0.3	0.2
.	1	0	0	0	0

This transition table will change from language to language due to language divergences.

Lexical Probability Table

	∈	people	laugh	…	…
^	1	0	0	…	0
N	0	$1x10^{-3}$	$1x10^{-5}$	…	…
V	0	$1x10^{-6}$	$1x10^{-3}$	…	…
O	0	0	0	…	…
.	1	0	0	0	0

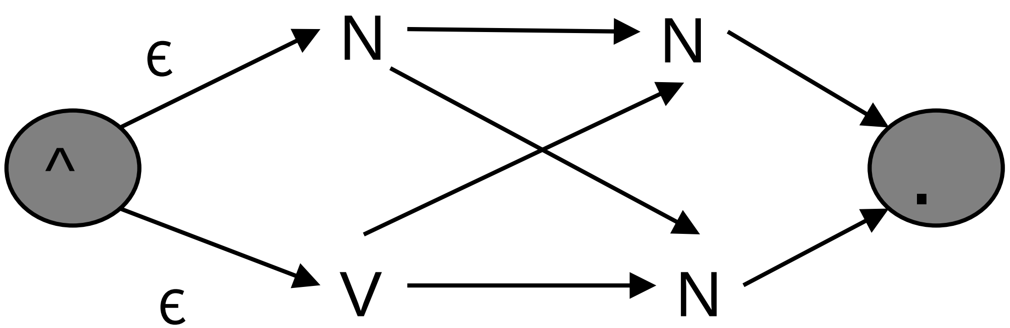
Size of this table = # pos tags in tagset X vocabulary size

vocabulary size = # unique words in corpus

Inference

New Sentence:

^ people laugh .



p(^ N N . | ^ people laugh .)

$= (0.6 \times 0.1) \times (0.1 \times 1 \times 10^{-3}) \times (0.2 \times 1 \times 10^{-5})$

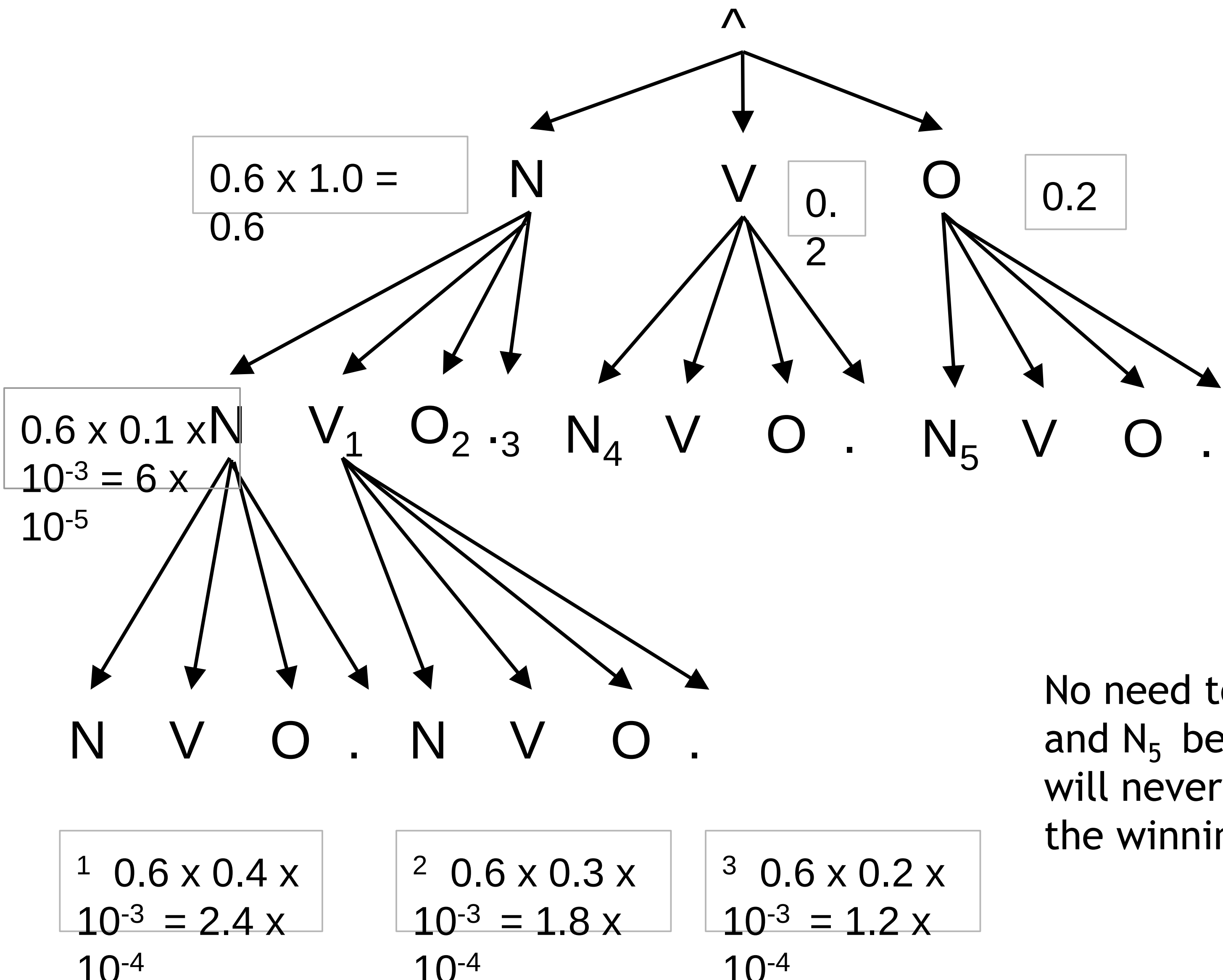
Computational Complexity

- If we have to get the probability of each sequence and then find maximum among them, we would run into exponential number of computations.
- If $|s|$ = #states (tags + ^ + .)

  and $|o|$ = length of sentence (words + ^ + .)

  Then, #sequences = $s^{|o|-2}$
- But, a large number of partial computations can be reused using Dynamic Programming.

Dynamic Programming



ϵ

people

laugh

No need to expand N_4 and N_5 because they will never be a part of the winning sequence.

Computational Complexity

- Retain only those N / V / O nodes which ends in the highest sequence probability.
- Now, complexity reduces from $|s|^{|o|}$ to $|s|.|o|$
- Here, we followed the Markov assumption of order 1.

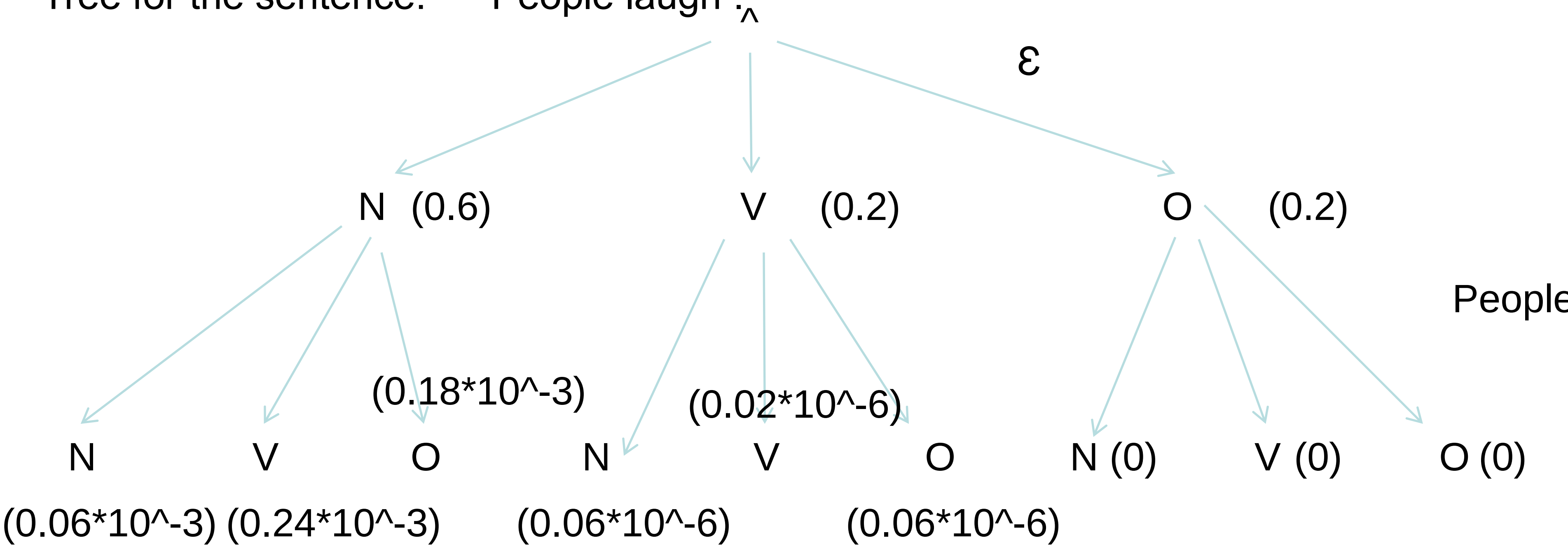
Points to ponder wrt HMM and Viterbi

Viterbi Algorithm

- Start with the start state.
- Keep advancing sequences that are “maximum” amongst all those ending in the same state

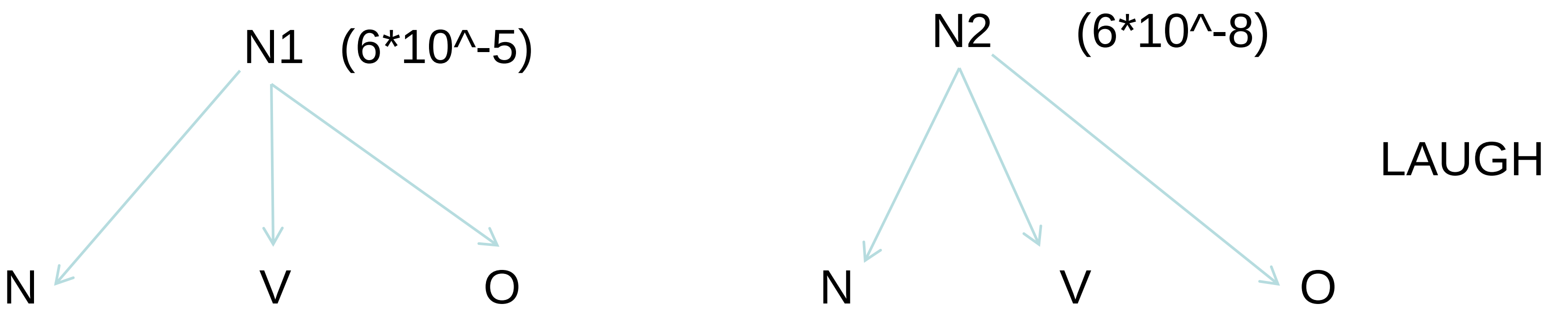
Viterbi Algorithm

Tree for the sentence: “^ People laugh .”



Claim: We do not need to draw all the subtrees in the algorithm

Viterbi phenomenon (Markov process)



Next step all the probabilities will be multiplied by identical probability (lexical and transition). So children of N2 will have probability less than the children of N1.

What does P(A|B) mean?

- P(A|B)= P(B|A)

  If P(A)=P(B)

- P(A|B) means??
  - Causality?? B causes A??
  - Sequentiality?? A follows B?

Reading List

- https://www.nltk.org/book/ch05.html
- TnT (http://www.aclweb.org/anthology-new/A/A00/A00-1031.pdf)
- Hindi POS Tagger built by IIT Bombay (http://www.cse.iitb.ac.in/pb/papers/ACL-2006-Hindi-POS-Tagging.pdf)