COS 484

Natural Language Processing

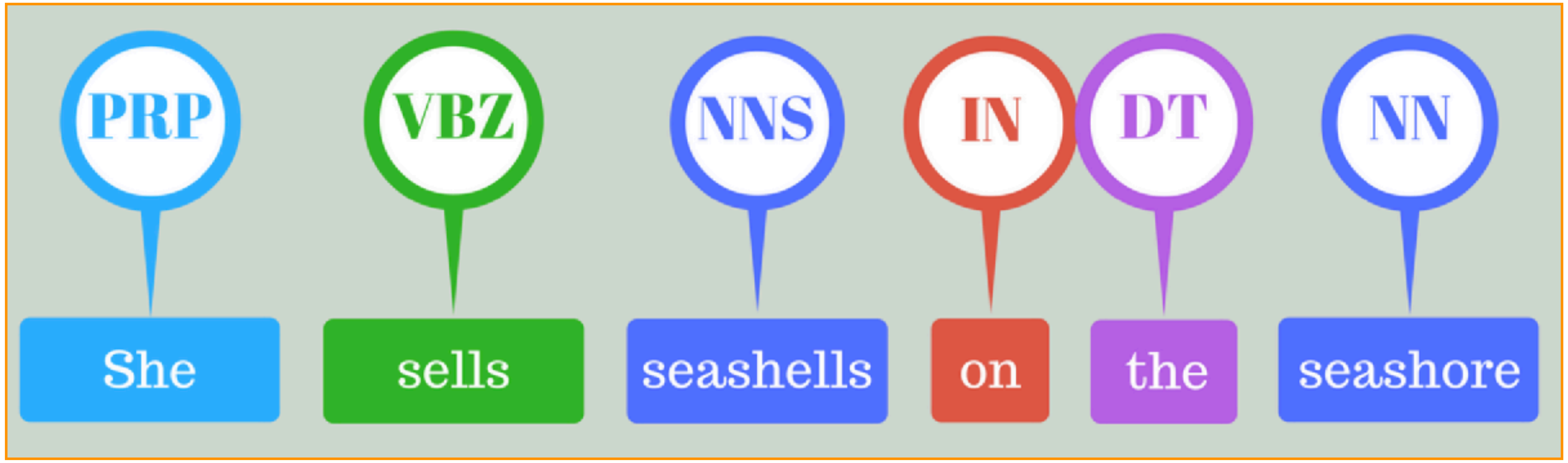
L6: Sequence Models

Spring 2025

Lecture plan

- Today:
  - Sequence tagging NLP tasks: part-of-speech tagging, named entity recognition
  - Hidden markov models
  - Viterbi algorithm
- Next lecture
  - Maximum entropy markov model (MEMMs)
  - Conditional random fields (CRFs)

Why model sequences?



Part-of-speech (POS) tagging

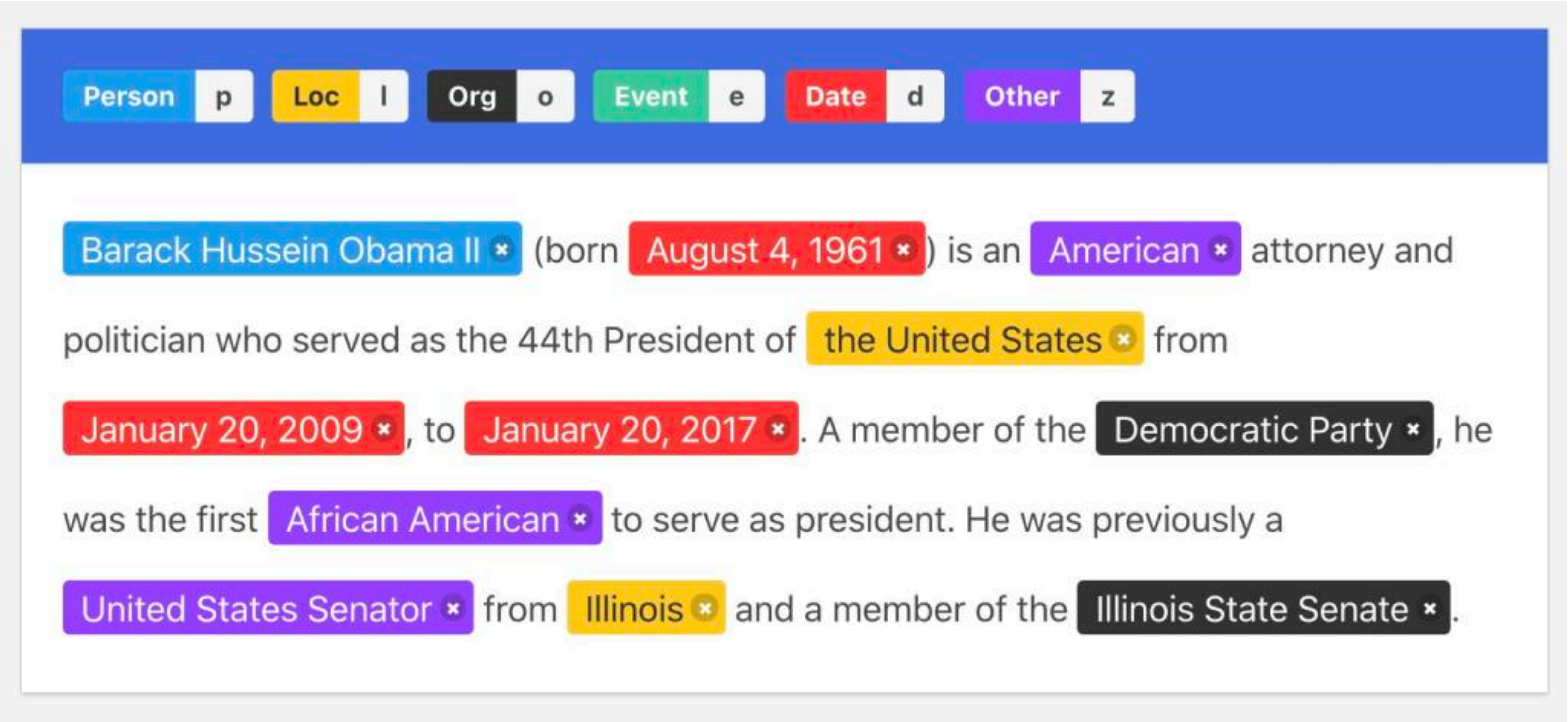
PRP: Personal pronoun

VBZ: Verb, 3rd person singular present

NN: singular noun
NNS: plural noun

IN: preposition or subordinating conjunction
DT: determiner

Why model sequences?

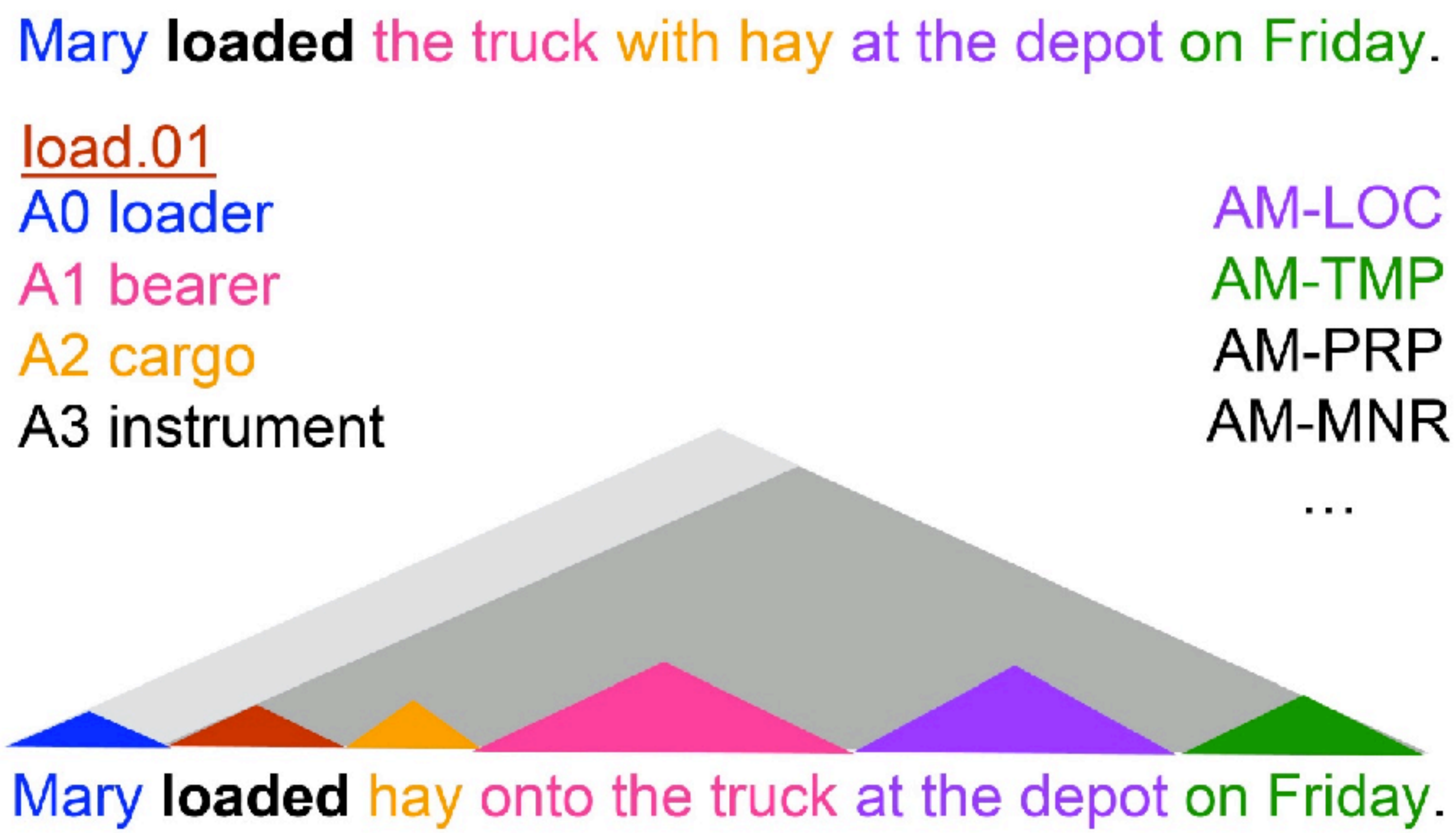


Barack Hussein Obama II (born August 4, 1961) is an American attorney and politician who served as the 44th President of the United States from January 20, 2009, to January 20, 2017. A member of the Democratic Party, he was the first African American to serve as president. He was previously a United States Senator from Illinois and a member of the Illinois State Senate.

Named Entity recognition

Image: https://www.analyticsvidhya.com/blog/2021/11/a-beginners-introduction-to-ner-named-entity-recognition/

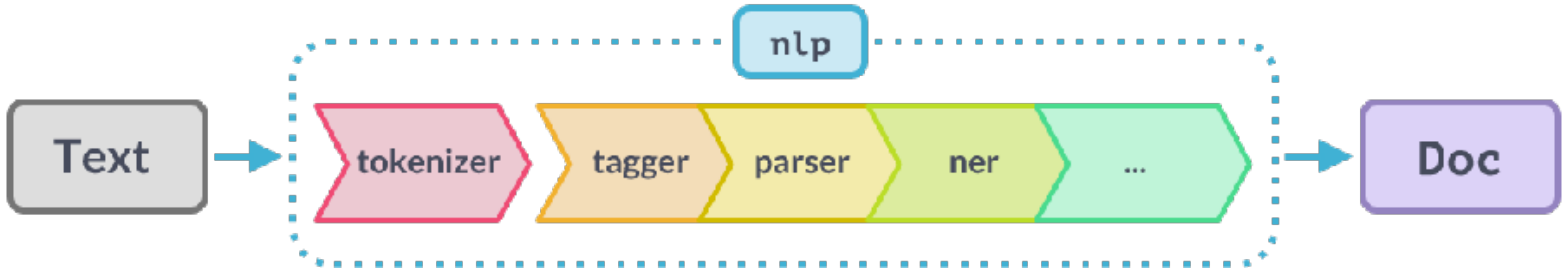
Why model sequences?

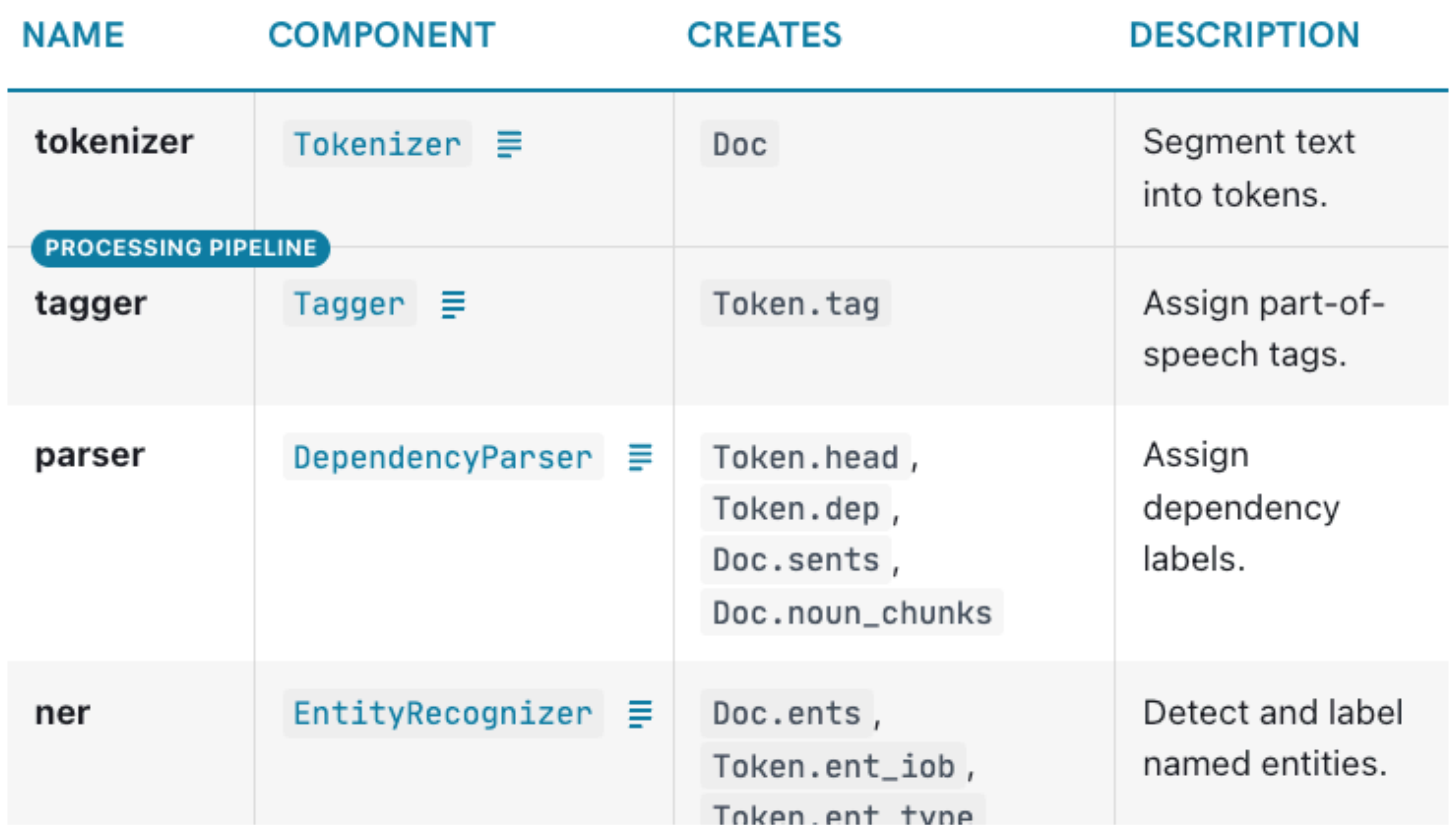

Mary loaded the truck with hay at the depot on Friday.
load.01
A0 loader
A1 bearer
A2 cargo
A3 instrument
AM-LOC
AM-TMP
AM-PRP
AM-MNR
...
Mary loaded hay onto the truck at the depot on Friday.

Semantic role labeling

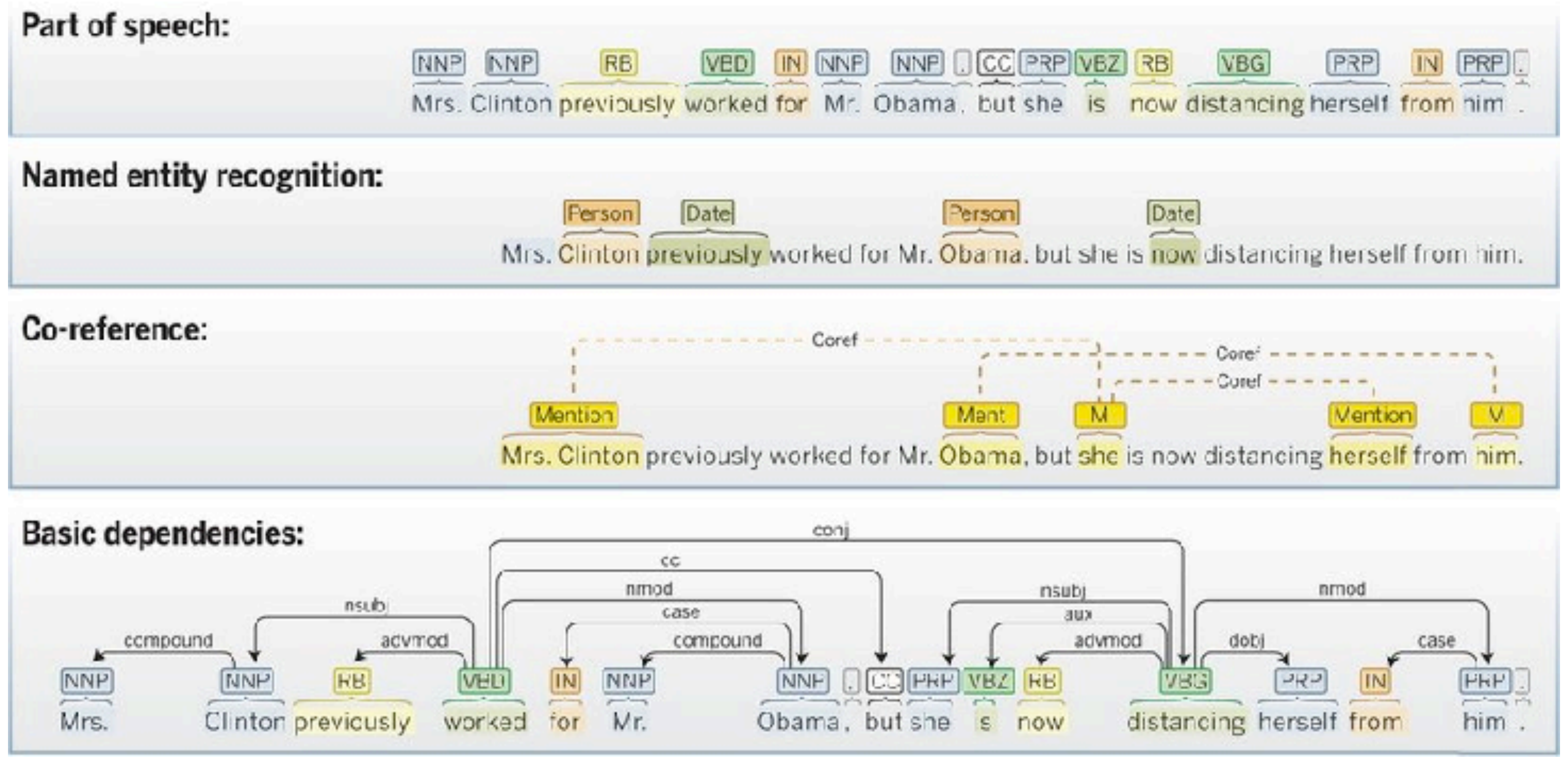
https://devopedia.org/semantic-role-labelling

NLP pipelines




PROCESSING PIPELINE

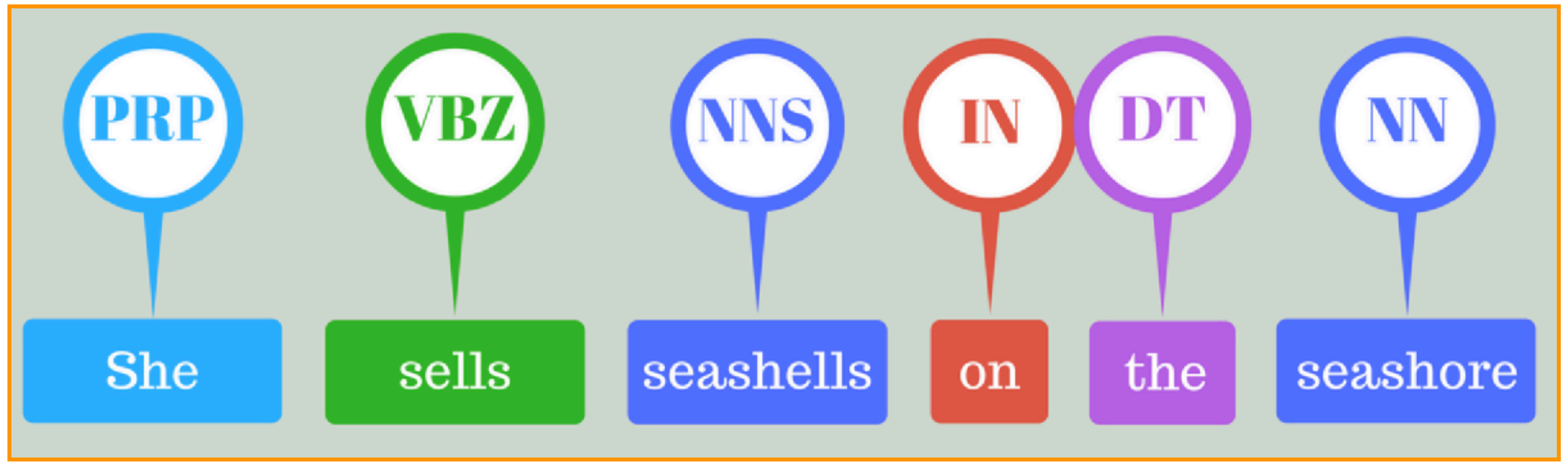
NAME	COMPONENT	CREATES	DESCRIPTION
tokenizer	Tokenizer	Doc	Segment text into tokens.
tagger	Tagger	Token.tag	Assign part-of-speech tags.
parser	DependencyParser	Token.head, Token.dep, Doc.sents, Doc.noun_chunks	Assign dependency labels.
ner	EntityRecognizer	Doc.ents, Token.ent_iob, Token.ent_type	Detect and label named entities.


Part of speech:
NNP NNP RB VBD IN NNP NNP . CC PRP VBZ RB VBG PRP IN PRP .
Mrs. Clinton previously worked for Mr. Obama, but she is now distancing herself from him .
Named entity recognition:
Person Date Person Date
Mrs. Clinton previously worked for Mr. Obama, but she is now distancing herself from him.
Co-reference:
Coref Coref Coref
Mention Ment M Mention M
Mrs. Clinton previously worked for Mr. Obama, but she is now distancing herself from him.
Basic dependencies:
conj cc nmod nsubj case compound advmod nsubj aux advmod dobj nmod case compound
NNP NNP RB VBD IN NNP NNP . CC PRP VBZ RB VBG PRP IN PRP .
Mrs. Clinton previously worked for Mr. Obama, but she is now distancing herself from him .

https://spacy.io/usage/processing-pipelines

https://stanfordnlp.github.io/CoreNLP/pipeline.html

What are part of speech tags?



- Word classes or syntactic categories
  - Reveal useful information about a word (and its neighbors!)

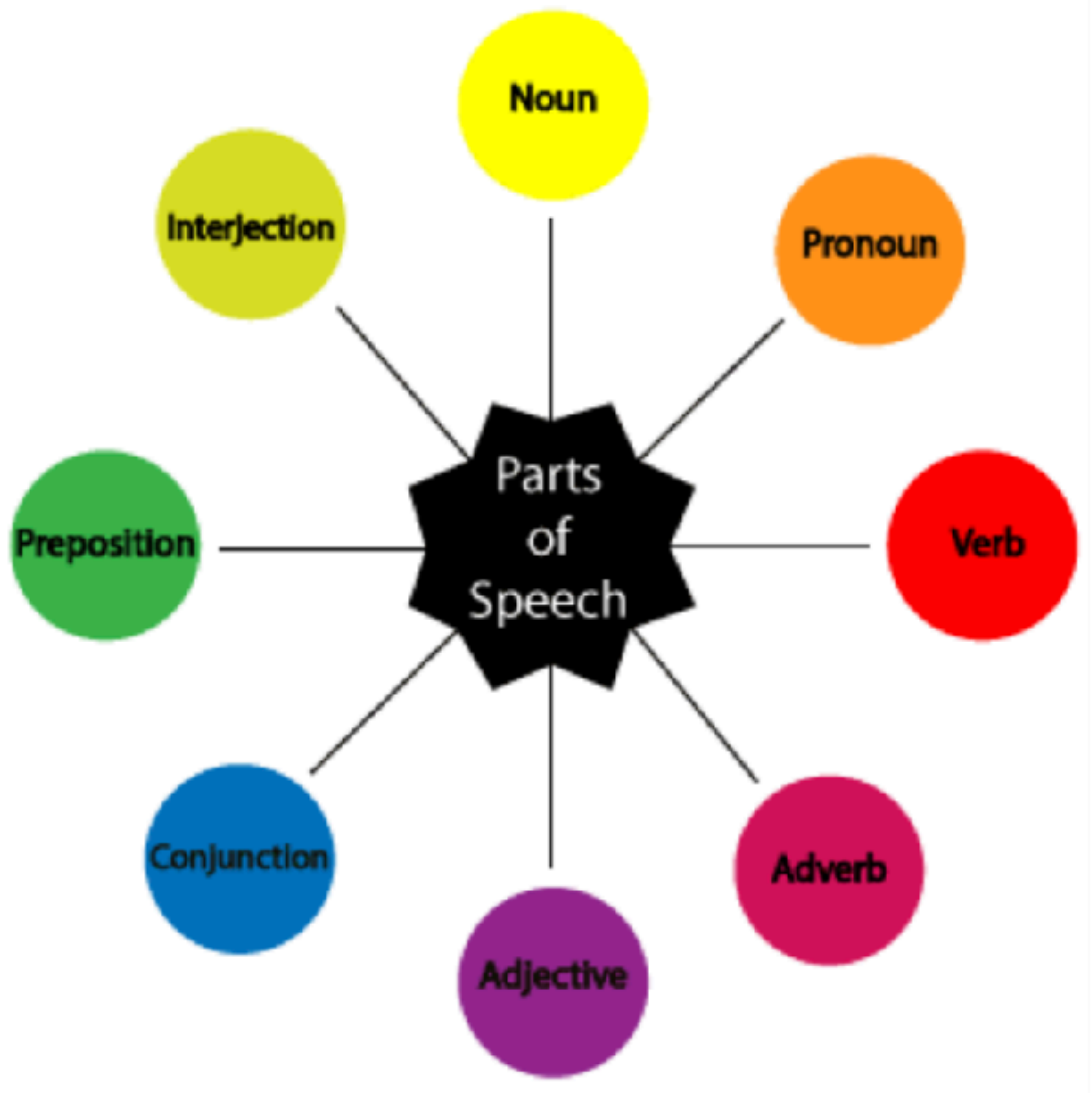
1. The/DT cat/NN sat/VBD on/IN the/DT mat/NN

2. Princeton/NNP is/VBZ in/IN New/NNP Jersey/NNP

3. The/DT old/NN man/VBP the/DT boat/NN

Parts of Speech

- Different words have different functions
- Can be roughly divided into two classes
- **Closed class**: fixed membership, **function words**
  - e.g. prepositions (*in, on, of*), determiners (*the, a*)
- **Open class**: New words get added frequently
  - e.g. nouns (Twitter, Facebook), verbs (google), adjectives, adverbs



Parts of Speech

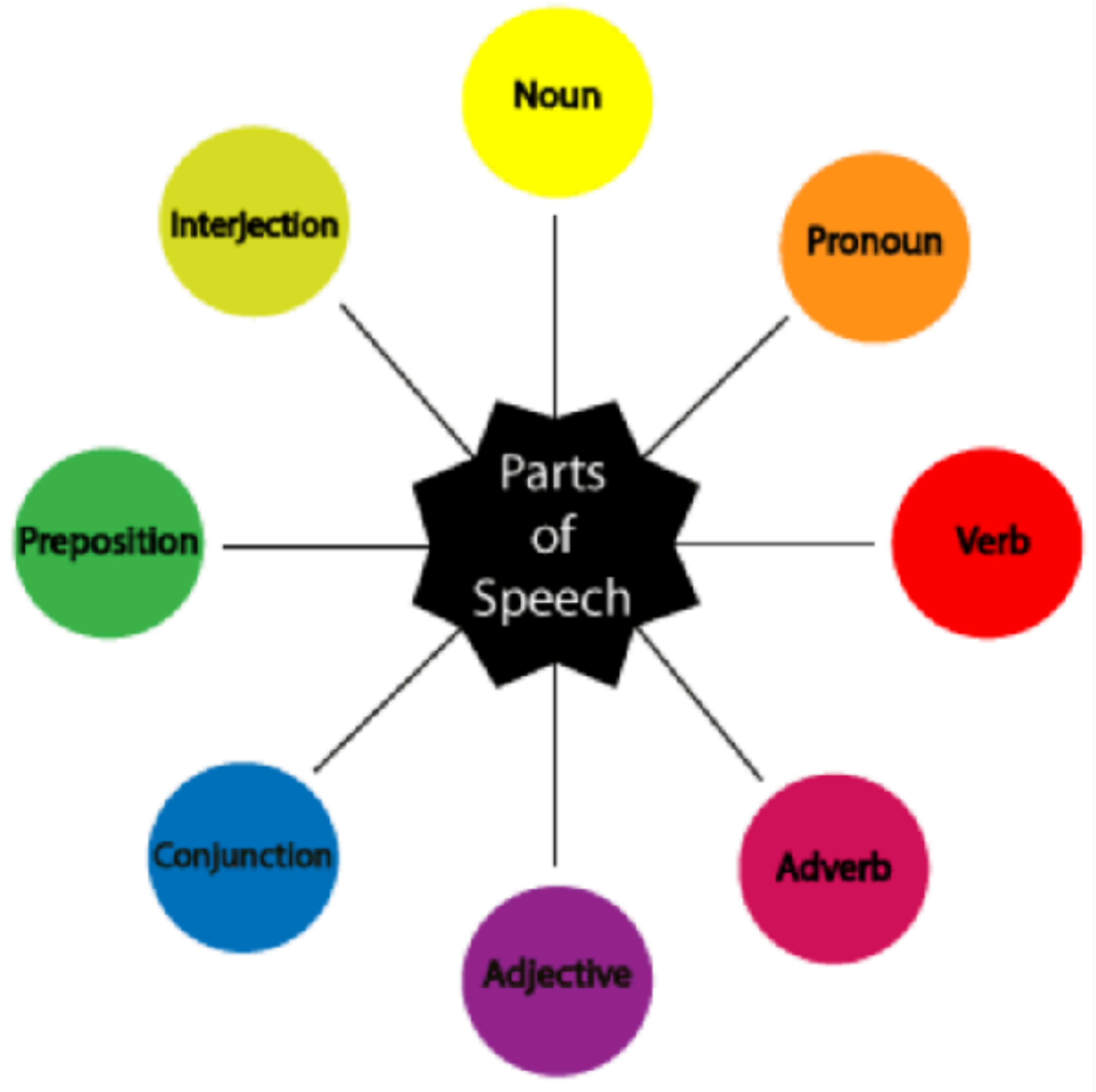
How many part of speech tags do you think English has?

A) < 10

B) 10 - 20

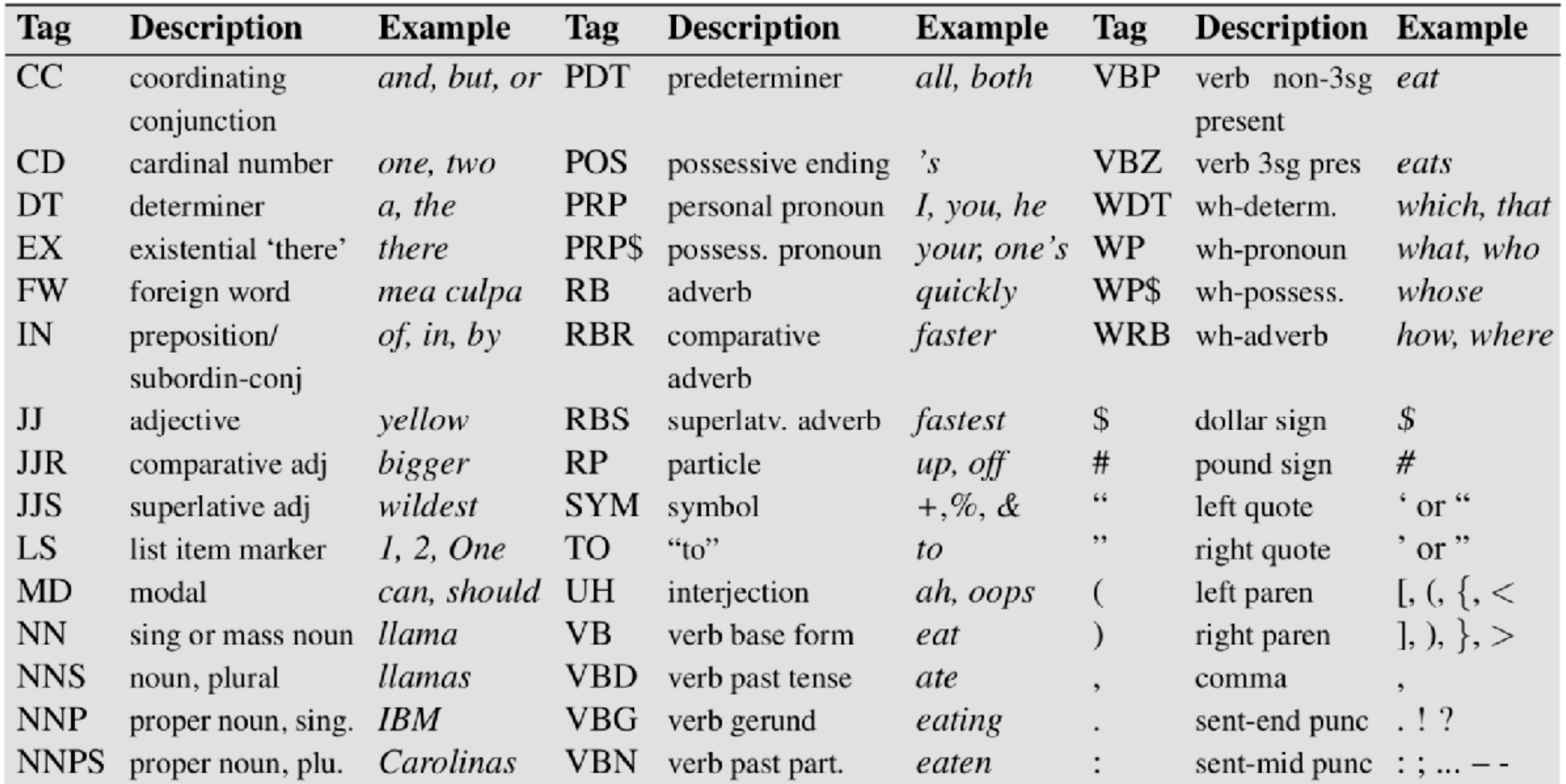
C) 20 - 40

D) > 40



The answer is (D) - well, depends on definitions!

Penn treebank part-of-speech tagset



Tag	Description	Example	Tag	Description	Example	Tag	Description	Example
CC	coordinating conjunction	*and, but, or*	PDT	predeterminer	*all, both*	VBP	verb non-3sg present	*eat*
CD	cardinal number	*one, two*	POS	possessive ending	*'s*	VBZ	verb 3sg pres	*eats*
DT	determiner	*a, the*	PRP	personal pronoun	*I, you, he*	WDT	wh-determ.	*which, that*
EX	existential ‘there’	*there*	PRP$	possess. pronoun	*your, one's*	WP	wh-pronoun	*what, who*
FW	foreign word	*mea culpa*	RB	adverb	*quickly*	WP$	wh-possess.	*whose*
IN	preposition/subordin-conj	*of, in, by*	RBR	comparative adverb	*faster*	WRB	wh-adverb	*how, where*
JJ	adjective	*yellow*	RBS	superlatv. adverb	*fastest*	$	dollar sign	*$*
JJR	comparative adj	*bigger*	RP	particle	*up, off*	#	pound sign	*#*
JJS	superlative adj	*wildest*	SYM	symbol	+,%, &	“	left quote	‘ or “
LS	list item marker	*1, 2, One*	TO	“to”	*to*	”	right quote	’ or ”
MD	modal	*can, should*	UH	interjection	*ah, oops*	(	left paren	[, (, {, <
NN	sing or mass noun	*llama*	VB	verb base form	*eat*	)	right paren	],), }, >
NNS	noun, plural	*llamas*	VBD	verb past tense	*ate*	,	comma	,
NNP	proper noun, sing.	*IBM*	VBG	verb gerund	*eating*	.	sent-end punc	. ! ?
NNPS	proper noun, plu.	*Carolinas*	VBN	verb past part.	*eaten*	:	sent-mid punc	: ; ... – -

45 tags

(Marcus et al., 1993)

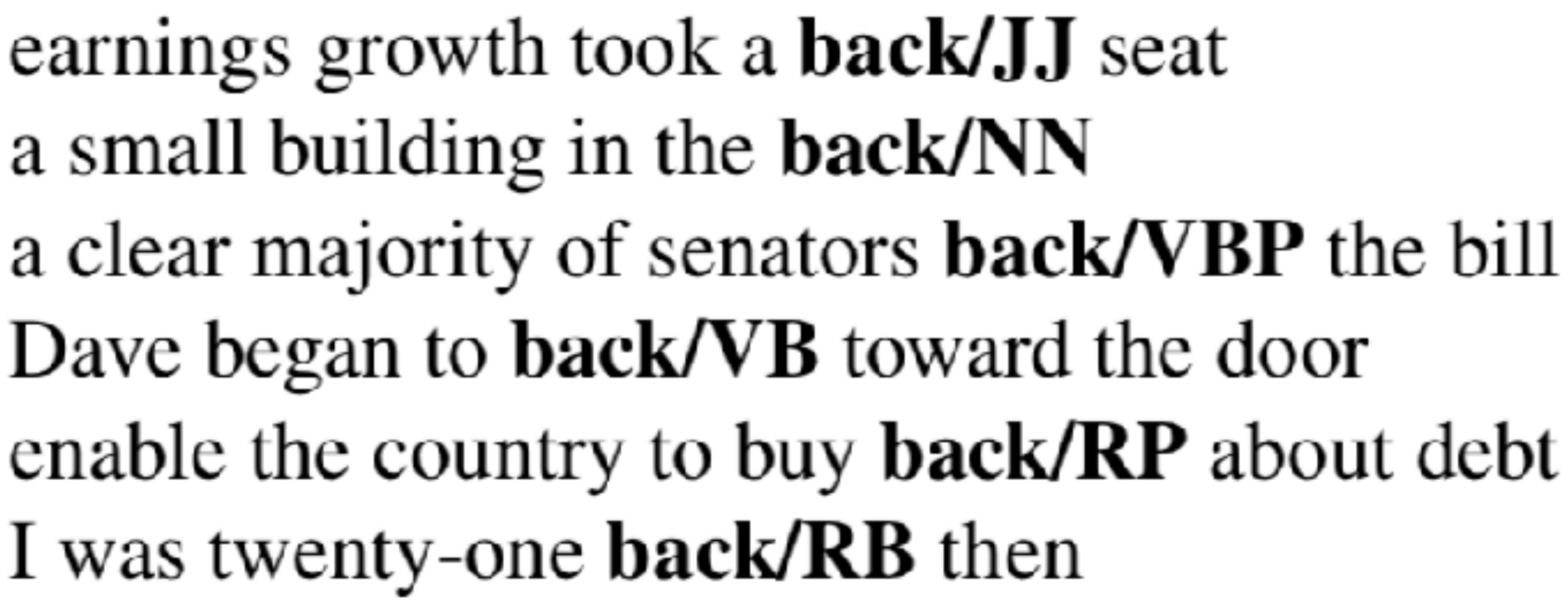
based on Wall Street Journal (WSJ)

Other corpora: Brown, Switchboard

Part of speech tagging

- Tag each word in a sentence with its part of speech
- Disambiguation task: each word might have different functions in different contexts
  - The/DT man/NN bought/VBD a/DT boat/NN
  - The/DT old/NN man/VBP the/DT boat/NN

Same word, different tags



earnings growth took a **back/JJ** seat
a small building in the **back/NN**
a clear majority of senators **back/VBP** the bill
Dave began to **back/VB** toward the door
enable the country to buy **back/RP** about debt
I was twenty-one **back/RB** then

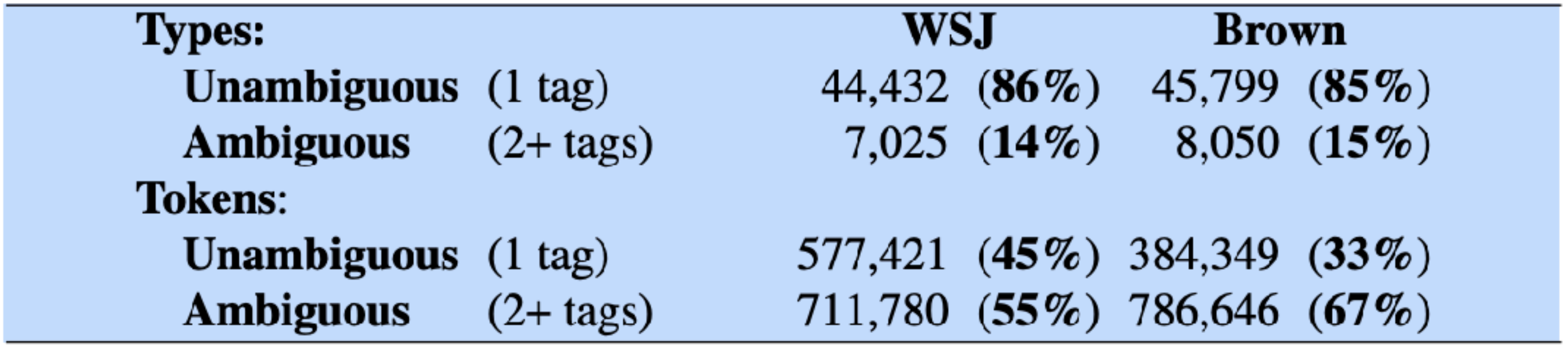
Some words have many functions!

JJ: adjective, NN: single or mass noun, VBP: Verb, non-3rd person singular present
VB: Verb, base form, RP: particle, RB: adverb

Part of speech tagging

- Tag each word in a sentence with its part of speech

- Disambiguation task: each word might have different senses/functions



Types:		**WSJ**		**Brown**	
Unambiguous	(1 tag)	44,432	(**86%**)	45,799	(**85%**)
Ambiguous	(2+ tags)	7,025	(**14%**)	8,050	(**15%**)
Tokens:					
Unambiguous	(1 tag)	577,421	(**45%**)	384,349	(**33%**)
Ambiguous	(2+ tags)	711,780	(**55%**)	786,646	(**67%**)

Unambiguous types:

Jane → NNP,

hesitantly → RB

- Types = distinct words in the corpus
- Tokens = all words in the corpus (can be repeated)

A simple baseline

- Most frequent class: Assign each word to the class it occurred most in the training set. (e.g. man/NN)
- How accurate do you think this baseline would be at tagging words?

  (A) <50%
  (B) 50-75%
  (C) 75-90%
  (D) >90%

A simple baseline

- Most frequent class: Assign each word to the class it occurred most in the training set. (e.g. man/NN)
- How accurate do you think this baseline would be at tagging words?

(A) <50%
(B) 50-75%
(C) 75-90%
(D) >90%

The answer is (D)

- This baseline accurately tags 92.34% of word tokens on Wall Street Journal (WSJ)!
- State of the art ~ 97% (also human-level acc)
- Average English sentence ~14 words
  - Sentence level accuracies: with 0.9214 per word is 31% vs 0.9714 per word is 65%
- POS tagging not solved yet!

How can we do better?

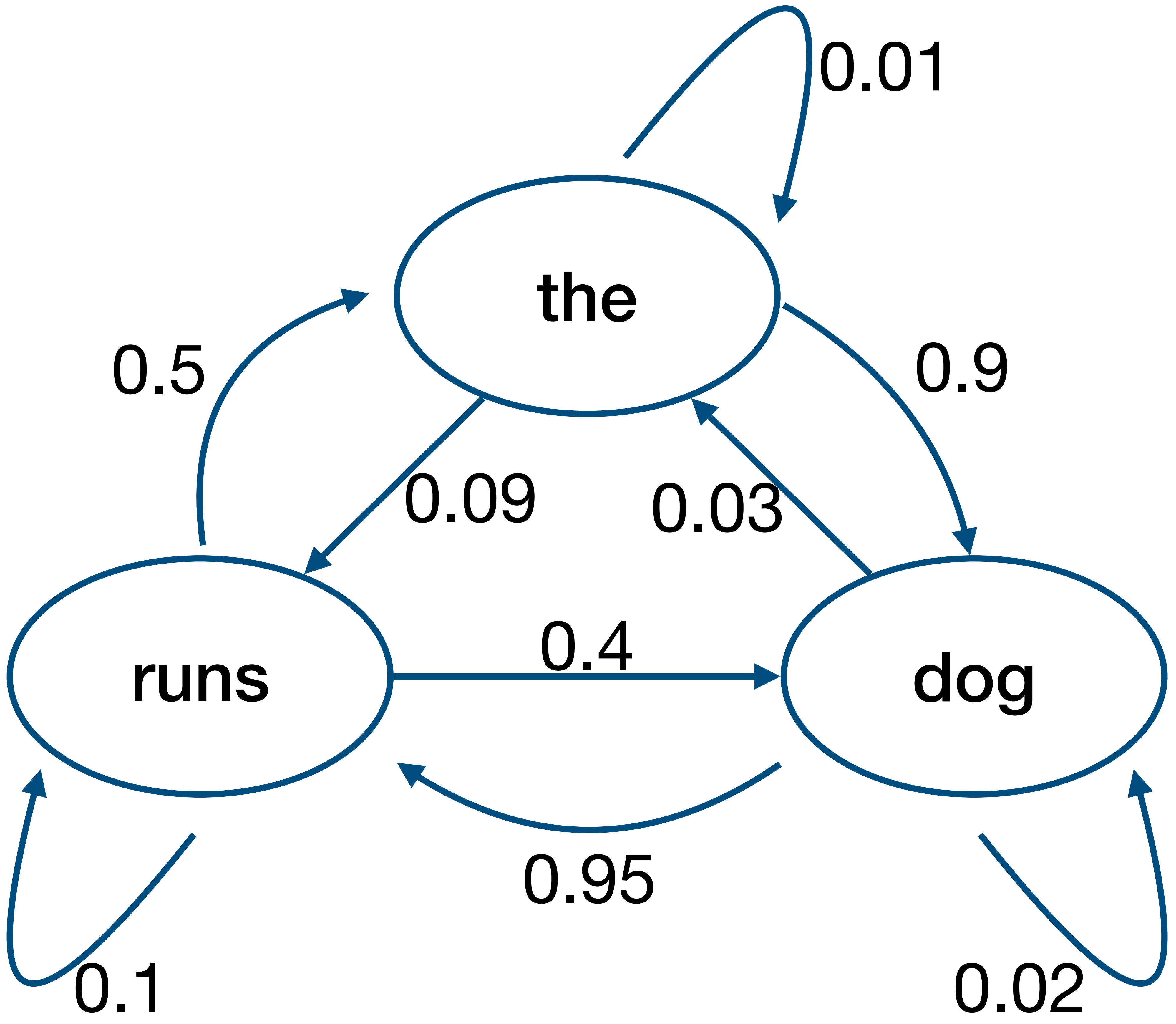
- The function (or POS) of a word depends on its context
  - The/DT old/JJ man/NN bought/VBP the/DT boat/NN
  - The/DT old/NN man/VBP the/DT boat/NN
- Certain POS combinations are extremely unlikely
  - *<JJ, DT>* ("good the") or *<DT, IN>* ("the in")
- Better to make decisions on entire sentences instead of individual words

Hidden Markov Models

Markov chains

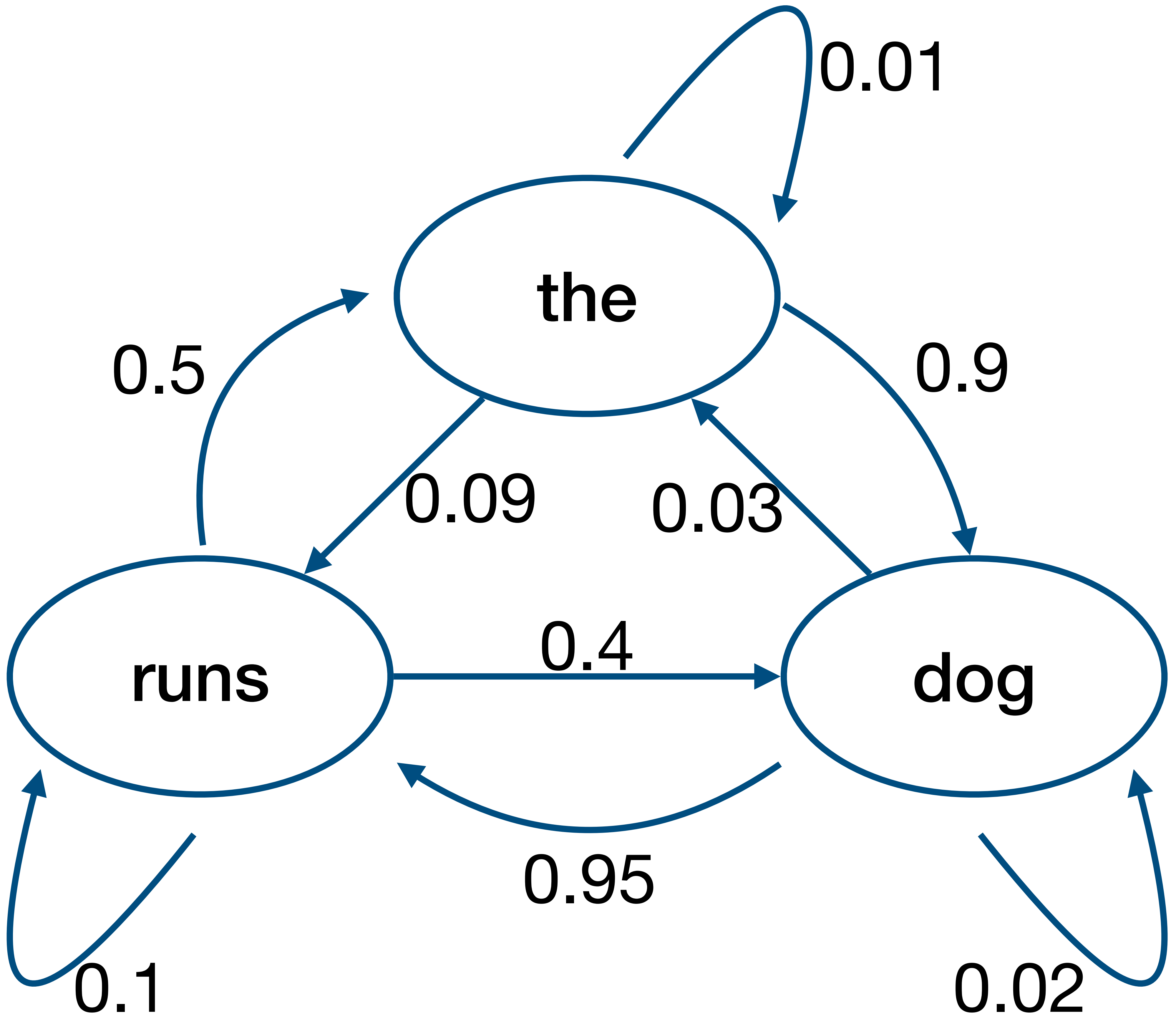
- Want to model the probability of difference sequences.
- Making an assumption that the next "state" only depends on current state.

Where have we seen this before?



Markov chains

- Each state can take one of K values (can assume {1, 2, ..., K} for simplicity)
- Markov assumption: $P(s_t \,|\, s_1, s_2, \ldots, s_{t-1}) \approx P(s_t \,|\, s_{t-1})$
- A Markov chain is specified by
  - Initial probability distribution $\pi(s), \forall s \in \{1, \ldots, K\}$
  - Transition probability matrix ($K \times K$)



Markov chains

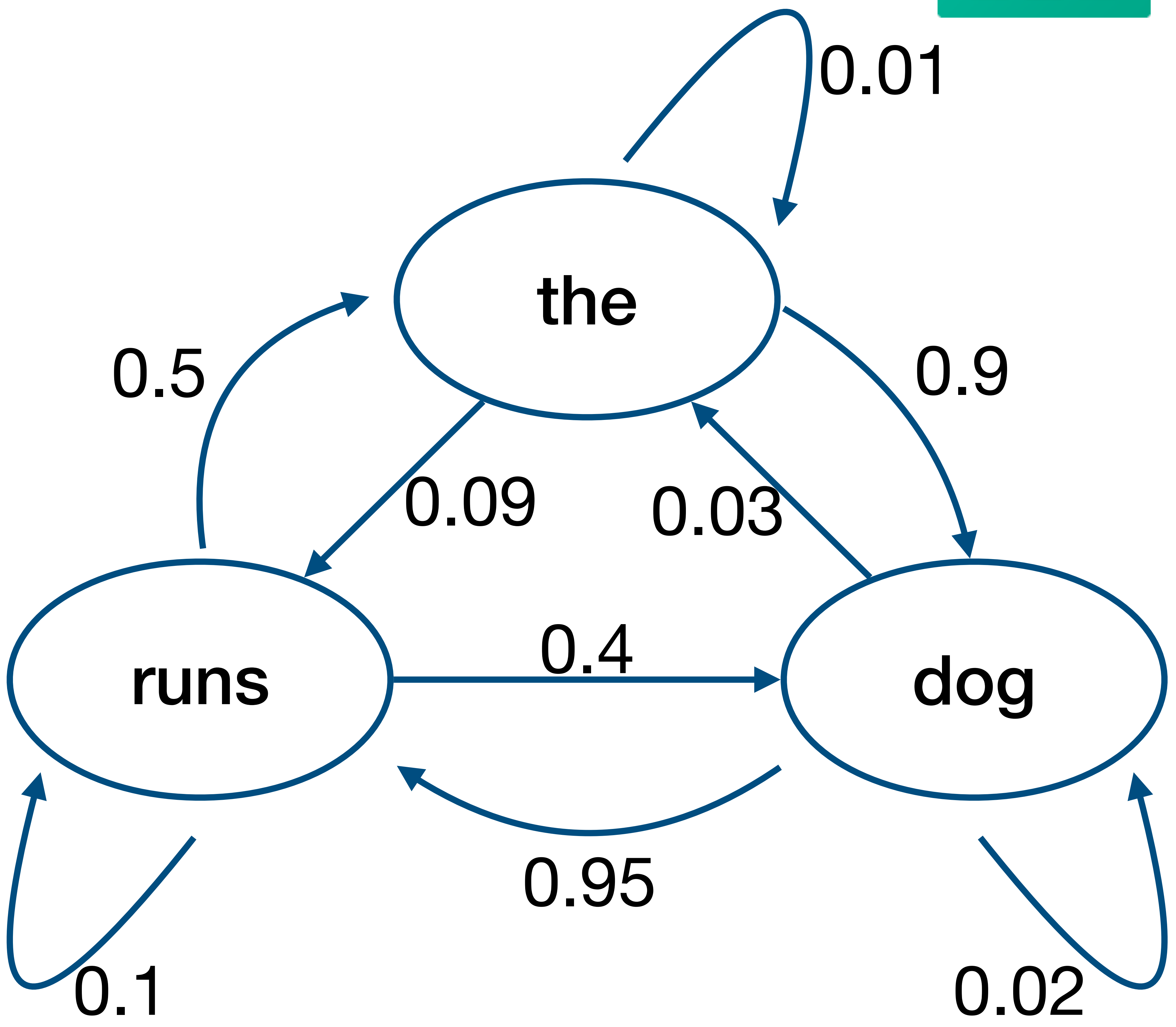
What is the probability of the sequence "the dog runs"? Assume $\pi(\text{"the"}) = 0.8$

(A) 0.8 x 0.9 x 0.95

(B) 0.8 x 0.99 x 0.98

(C) 0.2 x 0.9 x 0.95

(D) 0.2 x 0.01 x 0.02 x 0.1



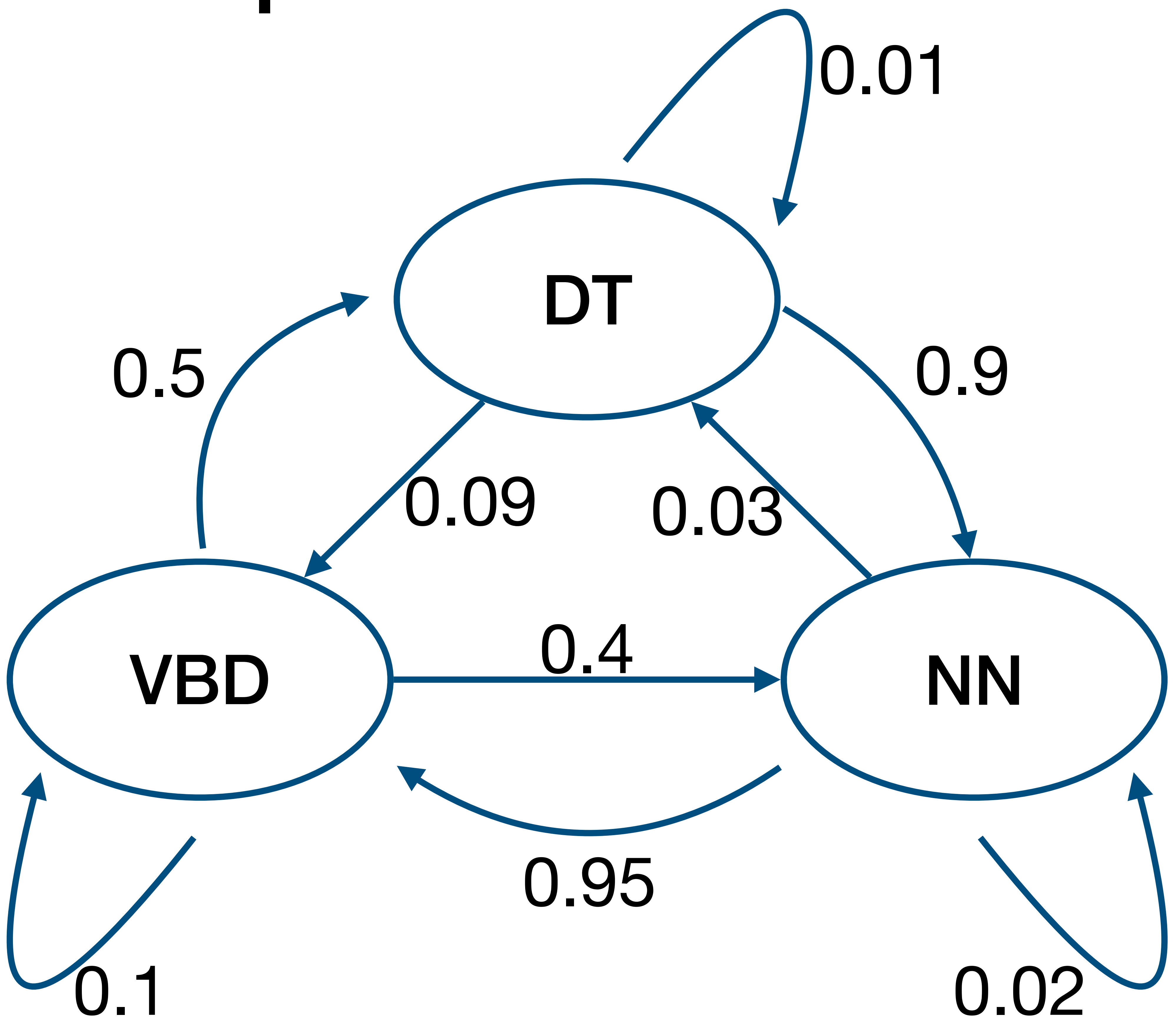
The answer is (A)

Markov chains for Part-of-speech

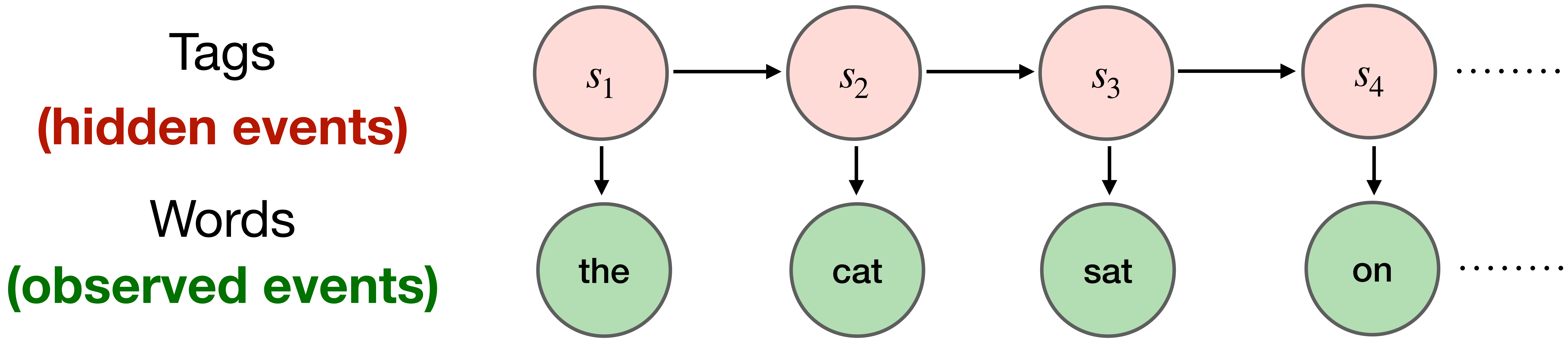
The/DT cat/NN sat/VBD on/IN the/DT mat/NN

- We want the states to be the part of speech tags.
- **Problem**: We don't normally see sequences of POS tags appearing in text:

  The/?? cat/?? sat/?? on/?? the/?? mat/??



Hidden Markov Model (HMM)



Tags
(hidden events)

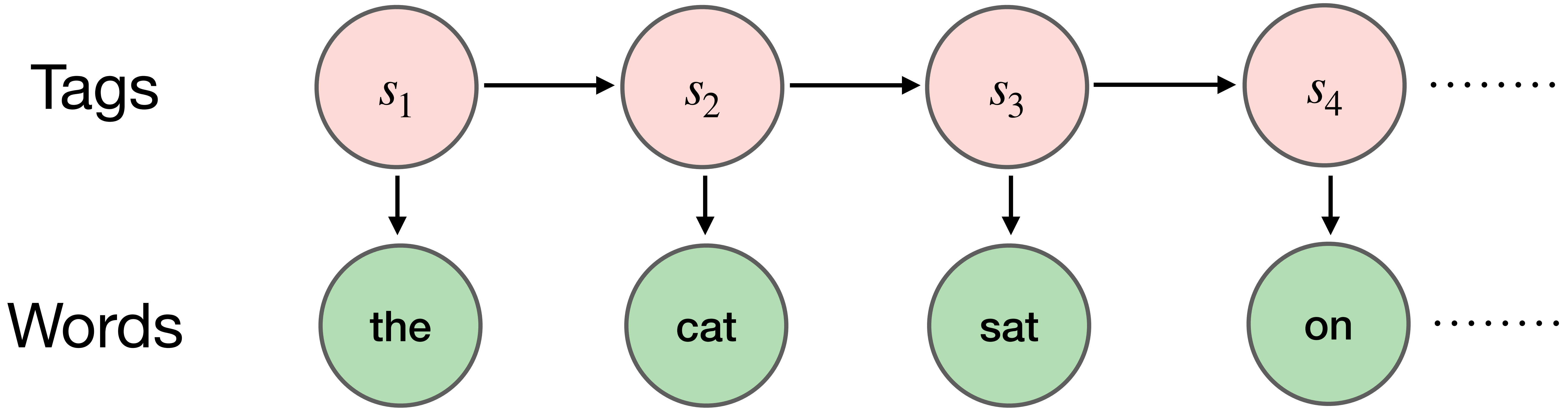
Words
(observed events)

s_1 s_2 s_3 s_4 ....

the cat sat on ....

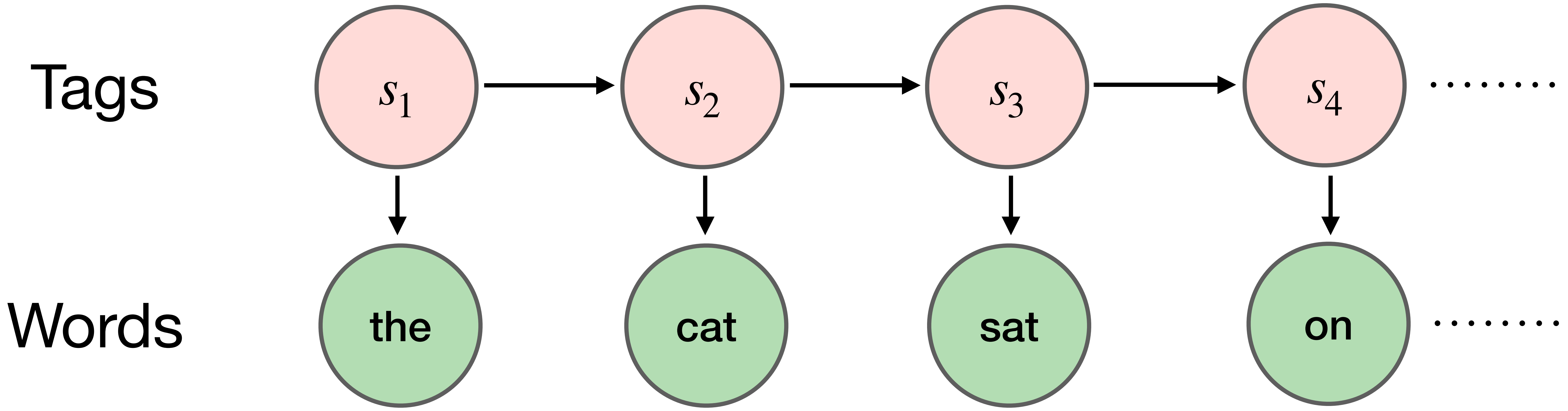
- We don't normally see sequences of POS tags in text
- However, we do observe the words!
- The HMM allows us to *jointly reason* over both **hidden** and **observed** events.
  - Assume that each position has a tag that generates a word

Components of an HMM



1. Set of states S = {1, 2, ..., K} and set of observations $O = \{o_1, \ldots, o_n\} \quad o_i \in V$
2. **Initial state probability distribution** $\pi(s_1)$
3. **Transition probabilities** $P(s_{t+1} \mid s_t)$
4. **Emission probabilities** $P(o_t \mid s_t)$

Assumptions



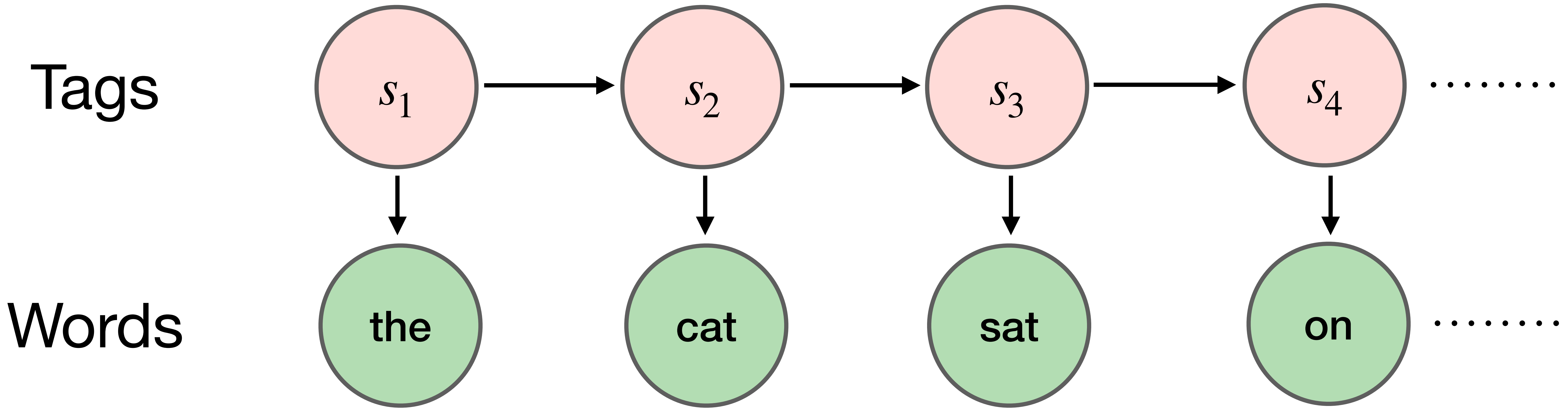
1. Markov assumption:

$$P(s_t \,|\, s_1, \ldots, s_{t-1}) \approx P(s_t \,|\, s_{t-1})$$

2. Output independence:

$$P(o_t \,|\, s_1, \ldots, s_t) \approx P(o_t \,|\, s_t)$$

Sequence likelihood



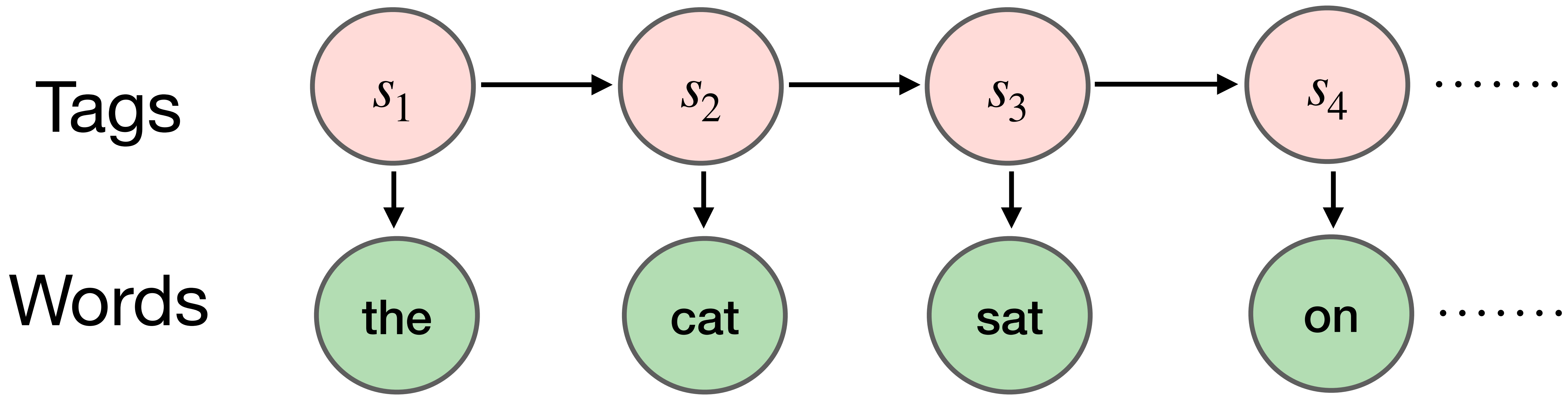
$$P(S,O) = P(s_1, s_2, \ldots, s_n, o_1, o_2, \ldots, o_n)$$

$$= \pi(s_1)p(o_1 \mid s_1)\prod_{i=2}^{n} \underbrace{P(s_i \mid s_{i-1})}_{\text{transition probability}}\underbrace{P(o_i \mid s_i)}_{\text{emission probability}}$$

If we add a dummy state $s_0 = \varnothing$ at the beginning,

$$P(S,O) = \prod_{i=1}^{n} P(s_i \mid s_{i-1})P(o_i \mid s_i) \quad [\,\pi(s_1) = P(s_1 \mid \varnothing)]$$

Example: Sequence likelihood



Dummy start state

s_{t+1}

s_t		DT	NN
	∅	0.8	0.2
	DT	0.2	0.8
	NN	0.3	0.7

o_t

s_t		the	cat
	DT	0.9	0.1
	NN	0.5	0.5

What is the joint probability
$P($*the cat*$,$ *DT NN*$)$?

(A) $(0.8 \times 0.8) \times (0.9 \times 0.5)$

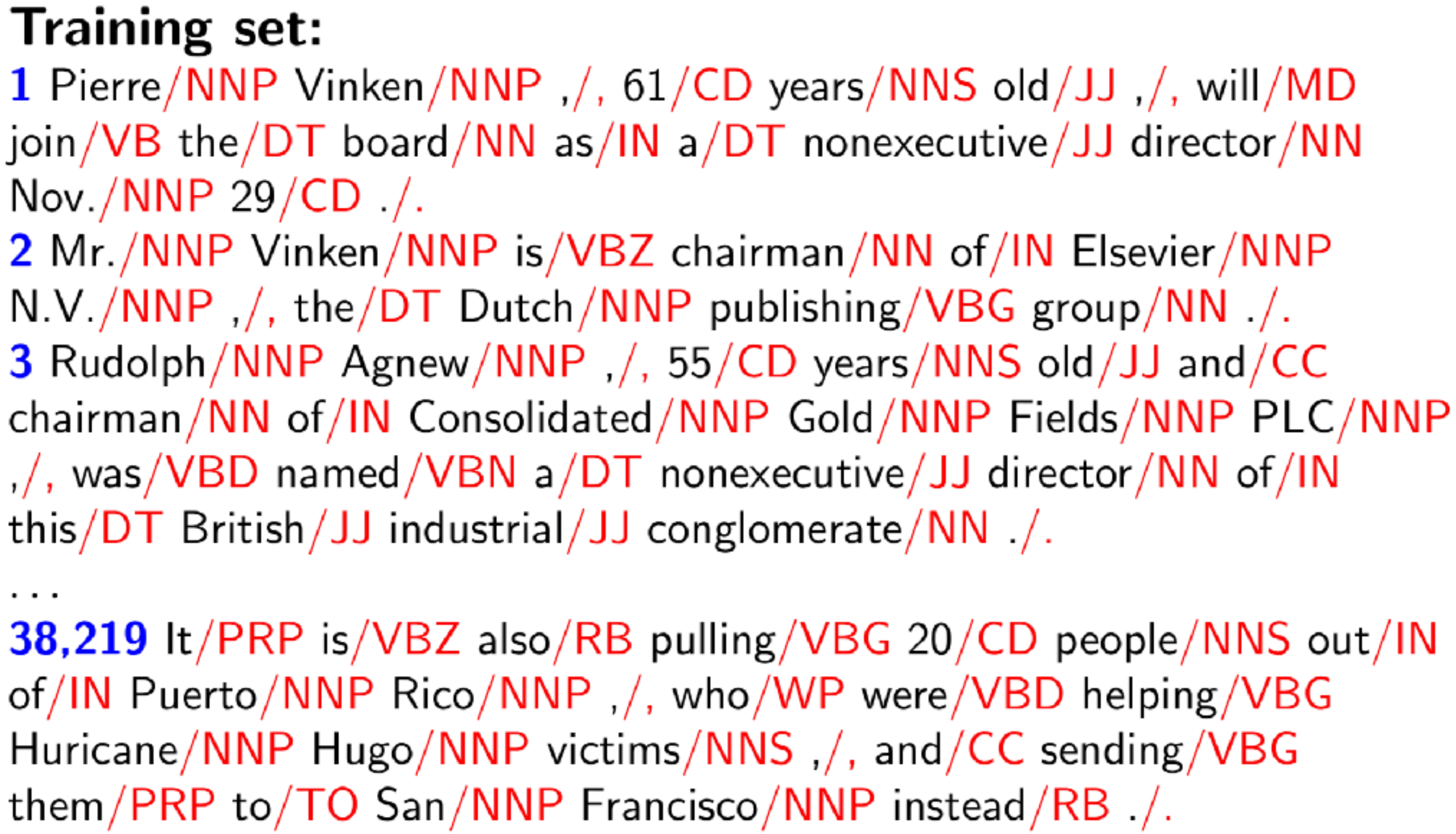
(B) $(0.2 \times 0.8) \times (0.9 \times 0.5)$

(C) $(0.3 \times 0.7) \times (0.5 \times 0.5)$

(D) $(0.8 \times 0.2) \times (0.5 \times 0.1)$

The answer is (A).

Learning



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 ./.

Maximum likelihood estimates:

$$P(s_i \mid s_j) = \frac{Count(s_j, s_i)}{Count(s_j)}$$

$$P(o \mid s) = \frac{Count(s, o)}{Count(s)}$$

Q: How many probabilities to estimate?

A: transition probabilities - (K + 1) × K

emission probabilities - $|V| \times K$

Learning example

Maximum likelihood estimates:

1. The/DT cat/NN sat/VBD on/IN the/DT mat/NN
2. Princeton/NNP is/VBZ in/IN New/NNP Jersey/NNP
3. The/DT old/NN man/VBP the/DT boat/NN

$$P(s_i \mid s_j) = \frac{Count(s_j, s_i)}{Count(s_j)}$$

$$P(o \mid s) = \frac{Count(s, o)}{Count(s)}$$

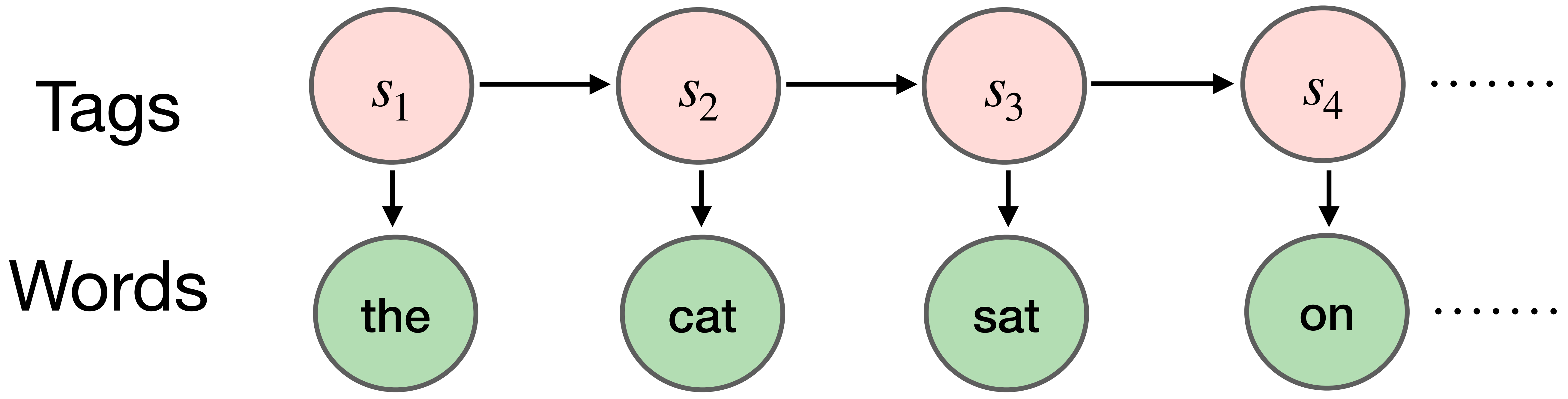
$\pi(DT) = P(DT \mid \varnothing) =$ 2/3

$P(NN \mid DT) =$ 4/4 $\quad$ $P(DT \mid IN) =$ 1/2

$P(cat \mid NN) =$ 1/4 $\quad$ $P(the \mid DT) =$ 2/4

(assuming we differentiate cased vs uncased words)

Decoding with HMMs



Task: Find the most probable sequence of states $S = s_1, s_2, \ldots, s_n$ given the observations $O = o_1, o_2, \ldots, o_n$

$$\hat{S} = \arg\max_S P(S \mid O) = \arg\max_S \frac{P(O \mid S)P(S)}{P(O)}$$ [Bayes' rule]

$$= \arg\max_S P(O \mid S)P(S)$$ [$P(O)$ doesn't depend on S!]

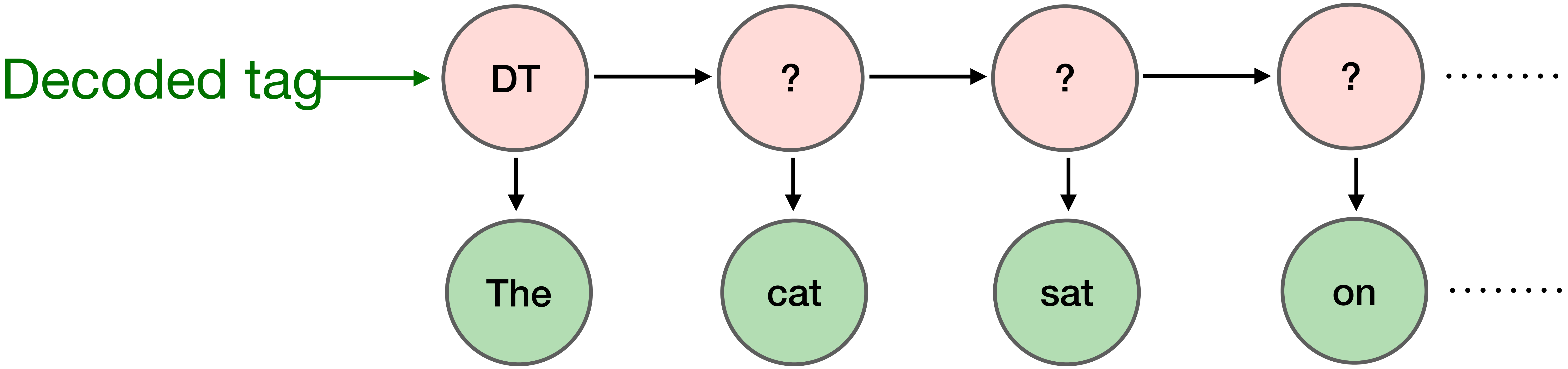
$$= \arg\max_{s_1, s_2, \ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1})$$ [Markov assumption]

How can we maximize this?
Search over all state sequences?

2 min stretch break

Greedy search

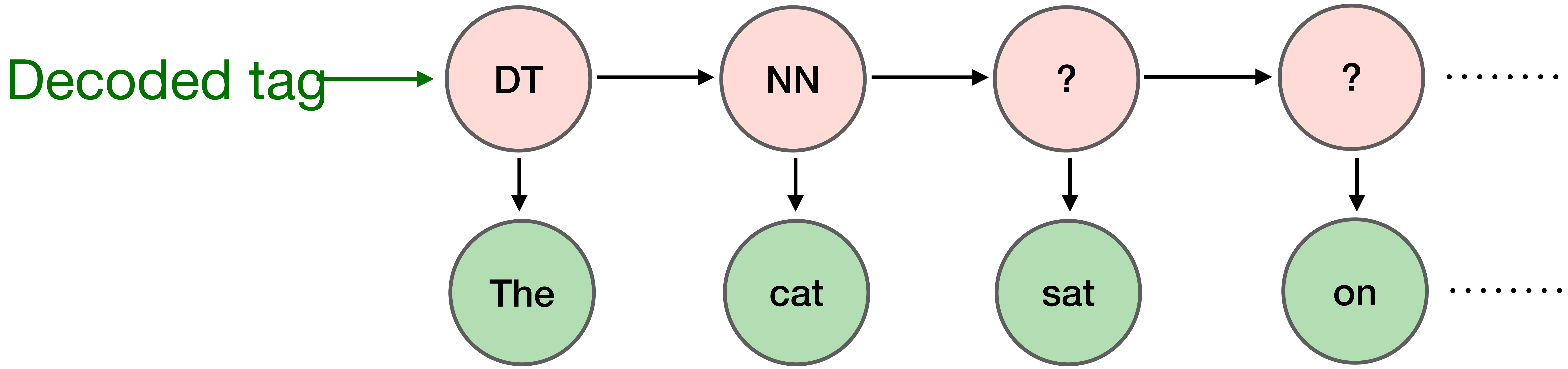
- Idea: Decode one state at at time



$$\arg\max_{s} \pi(s_1 = s) p(\text{The} \mid s) = \text{DT}$$

Greedy search

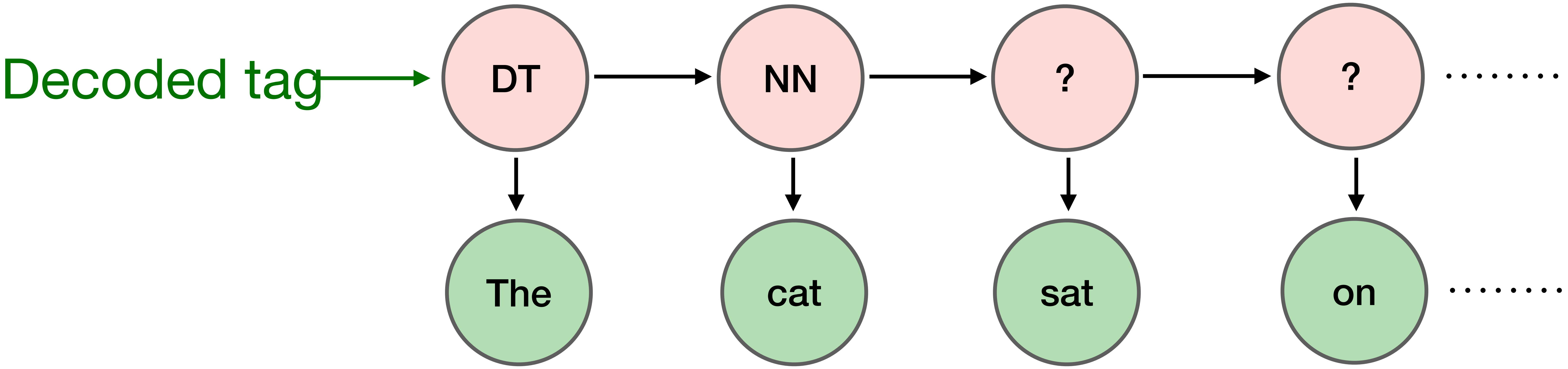
- Idea: Decode one state at at time



$$\arg\max_{s} p(s \mid DT) p(\text{cat} \mid s) = \text{NN}$$

Greedy search

- Idea: Decode one state at at time



- In general, $\hat{s}_t = \arg\max_{s} p(s \mid \hat{s}_{t-1}) p(o_t \mid s)$
- Very efficient, but not guaranteed to be optimum!

Viterbi decoding

- Use dynamic programming!
- Maintain some extra data structures
- Probability lattice, $M[T, K]$ and backtracking matrix, $B[T, K]$
  - T : Number of time steps
  - K : Number of states
- $M[i, j]$ stores joint probability of most probable sequence of states ending with state j at time i,
- $B[i, j]$ is the tag at time i-1 in the most probable sequence ending with tag j at time i

Viterbi decoding

- Recall: we want to compute $\hat{S} = \arg\max_{s_1, s_2, \ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1})$
- Let's first see how we can compute the maximum probability

$$\max_{s_1, \ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1}) = \max_{s_1, \ldots, s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \boxed{P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)}$$

These are the only terms that depend on s_1!

Slide adapted from Vivek Srikumar

Viterbi decoding

- Recall: we want to compute $\hat{S} = \arg \max_{s_1, s_2, \ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1})$

- Let's first see how we can compute the maximum probability

$$\max_{s_1, \ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1}) = \max_{s_1, \ldots, s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2, \ldots, s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \boxed{\cdot P(o_1 \mid s_1) \cdot P(s_1)}$$

Define
score$_1(s) = P(o_1 \mid s) \cdot P(s)$

Slide adapted from Vivek Srikumar

Viterbi decoding

- Recall: we want to compute $\hat{S} = \arg \max_{s_1, s_2, \ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1})$

- Let's first see how we can compute the maximum probability

$$\max_{s_1, \ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1}) = \max_{s_1, \ldots, s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2, \ldots, s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2, \ldots, s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot \text{score}_1(s_1)$$

Define
$\text{score}_1(s) = P(o_1 \mid s) \cdot P(s)$

Slide adapted from Vivek Srikumar

Viterbi decoding

$$\max_{s_1,\ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1}) = \max_{s_1,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot \text{score}_1(s_1)$$

Define

$\text{score}_1(s) = P(o_1 \mid s) \cdot P(s)$

Slide adapted from Vivek Srikumar

Viterbi decoding

$$\max_{s_1,\ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1}) = \max_{s_1,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot \text{score}_1(s_1)$$

$$= \max_{s_3,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdots P(o_3 \mid s_3) \boxed{\begin{array}{l} \max_{s_2} P(s_3 \mid s_2) \cdot P(o_2 \mid s_2) \cdot \\ \max_{s_1} P(s_2 \mid s_1) \cdot \text{score}_1(s_1) \end{array}}$$

Only terms that depend on s_2

Define

$\text{score}_1(s) = P(o_1 \mid s) \cdot P(s)$

Slide adapted from Vivek Srikumar

Viterbi decoding

$$\max_{s_1,\ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1}) = \max_{s_1,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot \text{score}_1(s_1)$$

$$= \max_{s_3,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdots P(o_3 \mid s_3) \max_{s_2} P(s_3 \mid s_2) \cdot \boxed{P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot \text{score}_1(s_1)}$$

Define

$$\text{score}_i(s) = \max_{s_{i-1}} \; P(s \mid s_{i-1}) \; P(o_i \mid s) \; \text{score}_{i-1}(s)$$

Slide adapted from Vivek Srikumar

Viterbi decoding

$$\max_{s_1,\ldots s_n} \prod_{i=1}^{n} P(o_i \mid s_i) P(s_i \mid s_{i-1}) = \max_{s_1,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot P(o_1 \mid s_1) \cdot P(s_1)$$

$$= \max_{s_2,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdot \cdots P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot \text{score}_1(s_1)$$

$$= \max_{s_3,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdots P(o_3 \mid s_3) \max_{s_2} P(s_3 \mid s_2) \cdot \boxed{P(o_2 \mid s_2) \cdot \max_{s_1} P(s_2 \mid s_1) \cdot \text{score}_1(s_1)}$$

$$= \max_{s_3,\ldots,s_n} P(o_n \mid s_n) \cdot P(o_{n-1} \mid s_{n-1}) \cdots P(o_3 \mid s_3) \max_{s_2} P(s_3 \mid s_2) \cdot \text{score}_2(s_2)$$

Define

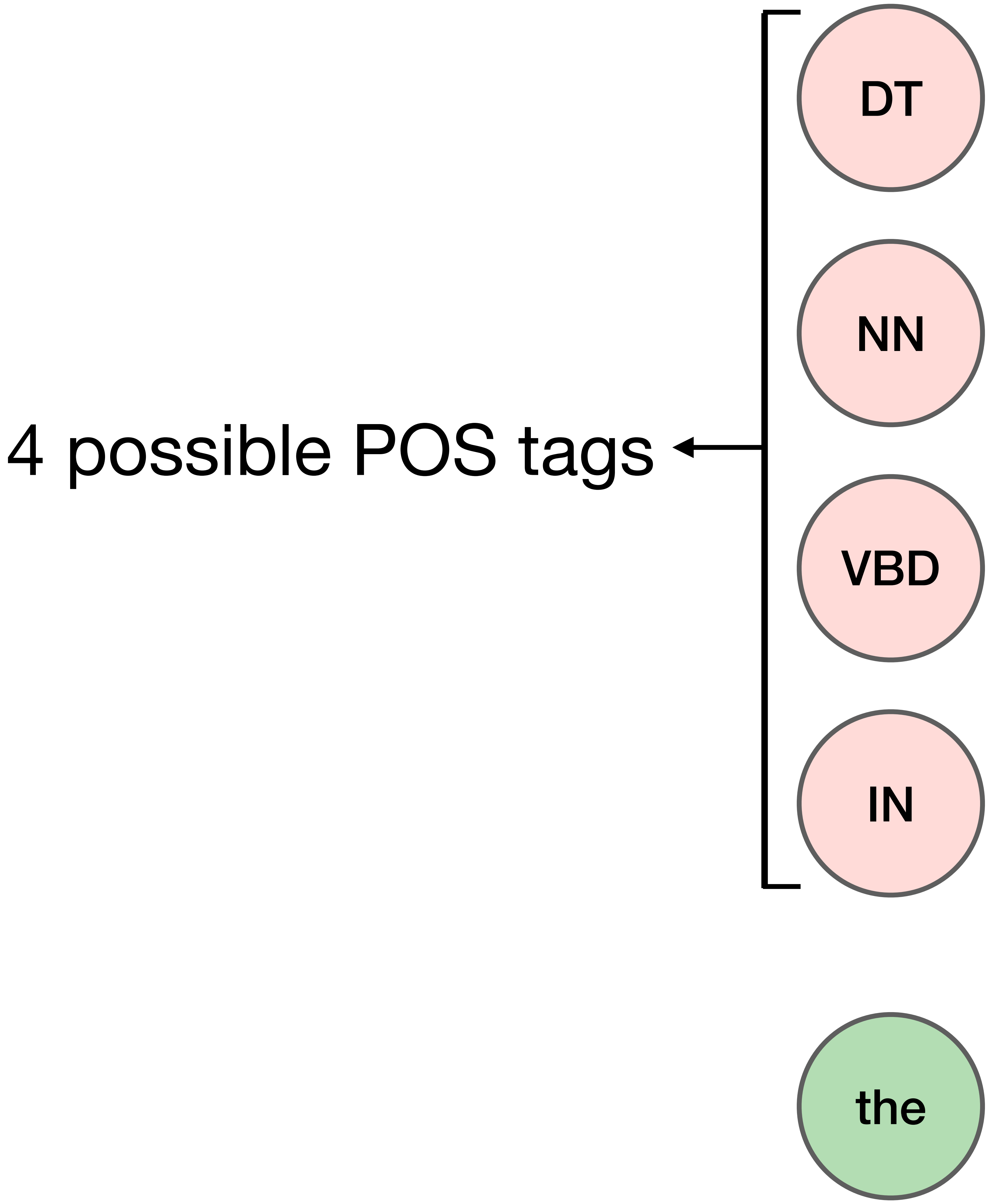
$$\text{score}_i(s) = \max_{s_{i-1}} \; P(s \mid s_{i-1}) \; P(o_i \mid s) \; \text{score}_{i-1}(s)$$

Slide adapted from Vivek Srikumar

Viterbi decoding

- Use dynamic programming!
- Maintain some extra data structures
- Probability lattice, $M[T, K]$ and backtracking matrix, $B[T, K]$
  - T : Number of time steps
  - K : Number of states
- $M[i, j]$ stores joint probability of most probable sequence of states ending with state j at time i,
- $B[i, j]$ is the tag at time i-1 in the most probable sequence ending with tag j at time i

Viterbi decoding example



$$M[1,DT] = \pi(DT)\ P(\text{the} \,|\, DT)$$

$$M[1,NN] = \pi(NN)\ P(\text{the} \,|\, NN)$$

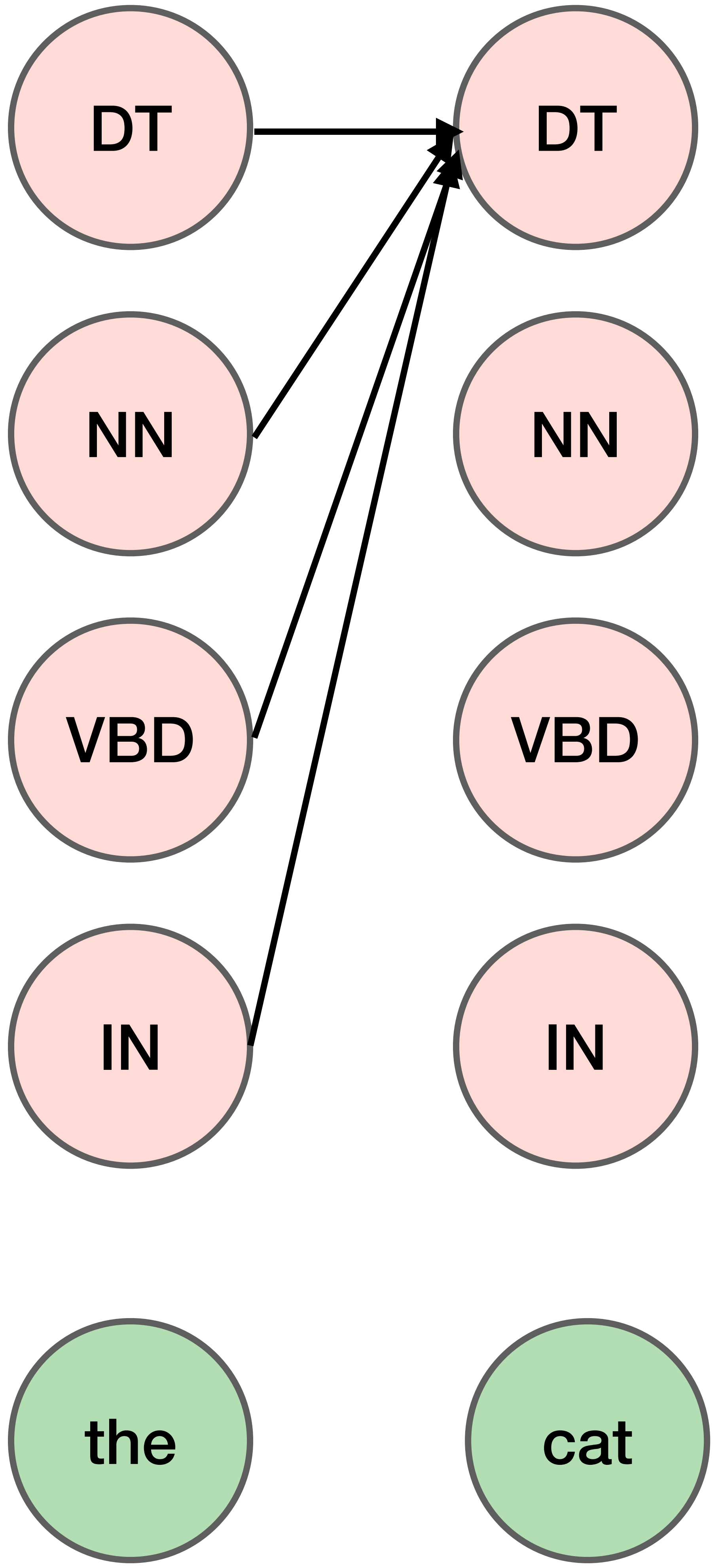
$$M[1,VBD] = \pi(VBD)\ P(\text{the} \,|\, VBD)$$

$$M[1,IN] = \pi(IN)\ P(\text{the} \,|\, IN)$$

Initialize the table: We store score$_1(s) = P(o_1 \,|\, s) \cdot P(s)$ in table $M[1, :]$

Forward

Viterbi decoding



$$M[2,DT] = \max_k M[1,k]\ P(DT\,|\,k)\ P(\text{cat}\,|\,DT)$$

$$M[2,NN] = \max_k M[1,k]\ P(NN\,|\,k)\ P(\text{cat}\,|\,NN)$$

$$M[2,VBD] = \max_k M[1,k]\ P(VBD\,|\,k)\ P(\text{cat}\,|\,VBD)$$

$$M[2,IN] = \max_k M[1,k]\ P(IN\,|\,k)\ P(\text{cat}\,|\,IN)$$

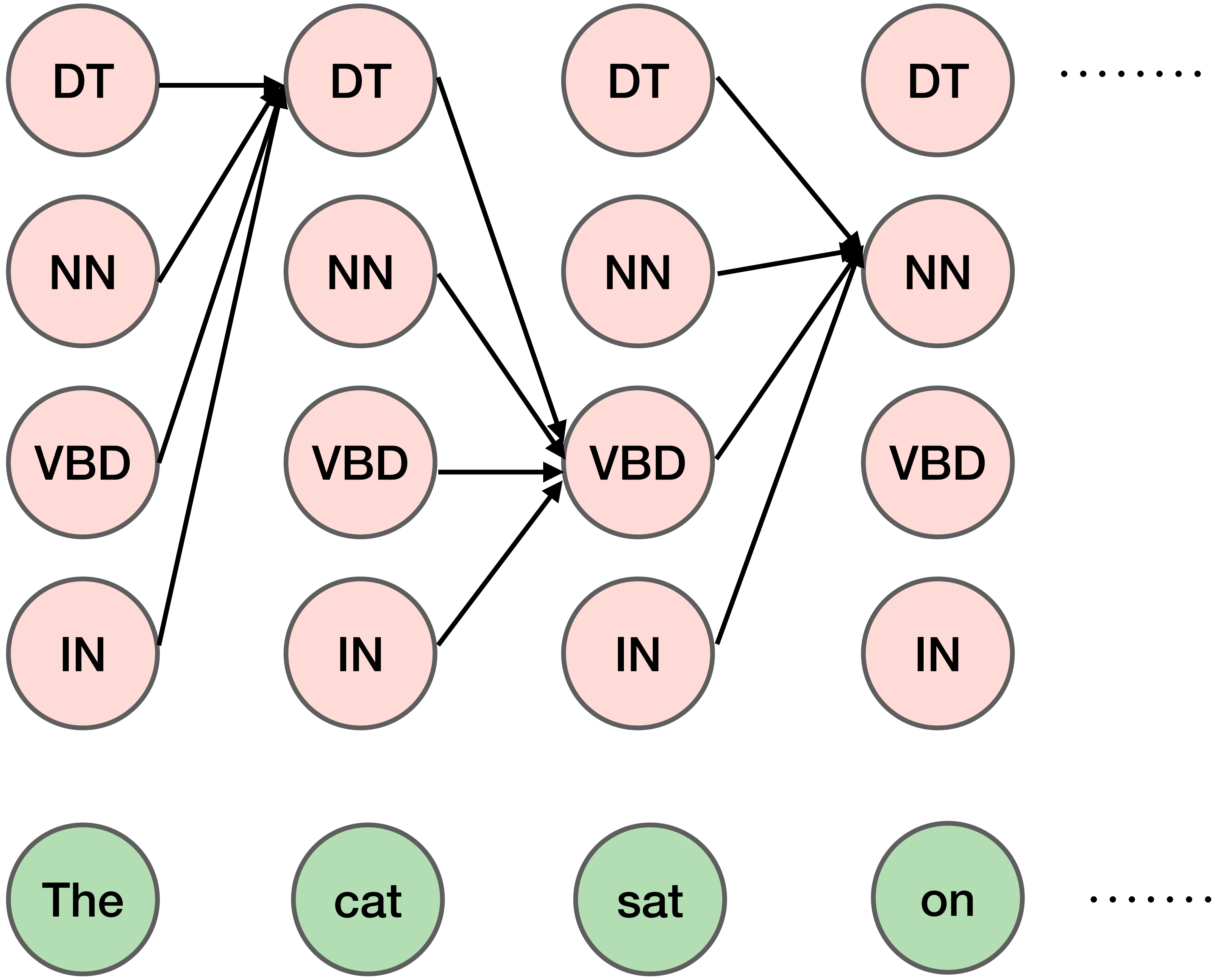
Next: We store

$\text{score}_2(s) = \max_{s_1} P(s\,|\,s_1) \cdot P(o_2\,|\,s) \cdot M[1,s_1]$

in table $M[2, :]$

Forward

Viterbi decoding



What is the time complexity of this algorithm? Let n be the number of time steps (length of the sequence), and K be the number of states.

(A) $O(n)$

(B) $O(nK)$

(C) $O(nK^2)$

(D) $O(n^2K)$

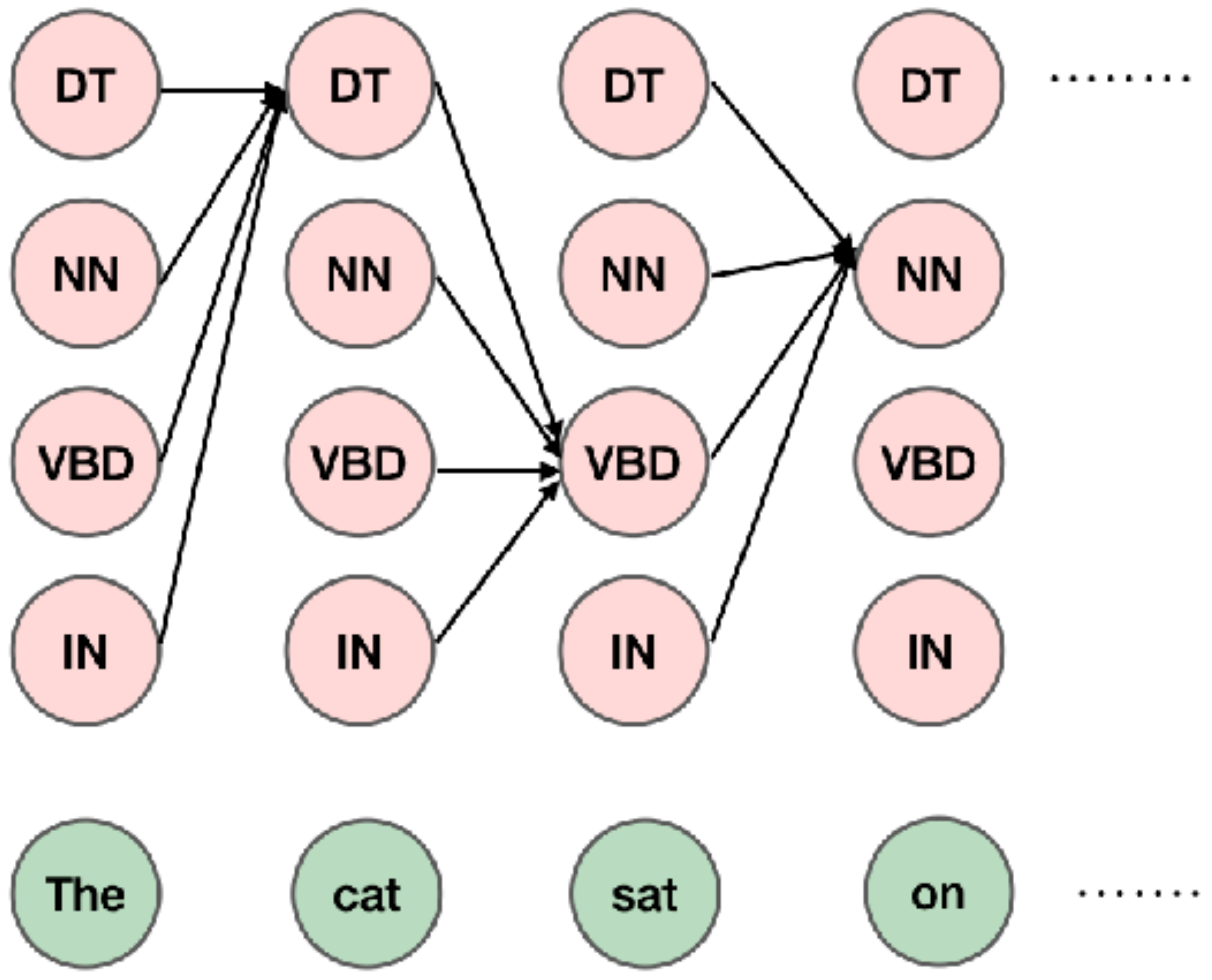
The answer is (C).

In general:

$$M[i, j] = \max_k M[i-1, k]\ P(s_j \mid s_k)\ P(o_i \mid s_j) \quad 1 \leq k \leq K \ \ 1 \leq i \leq n$$

Viterbi decoding

Backward: Pick $\max_k M[n, k]$ and backtrack using B



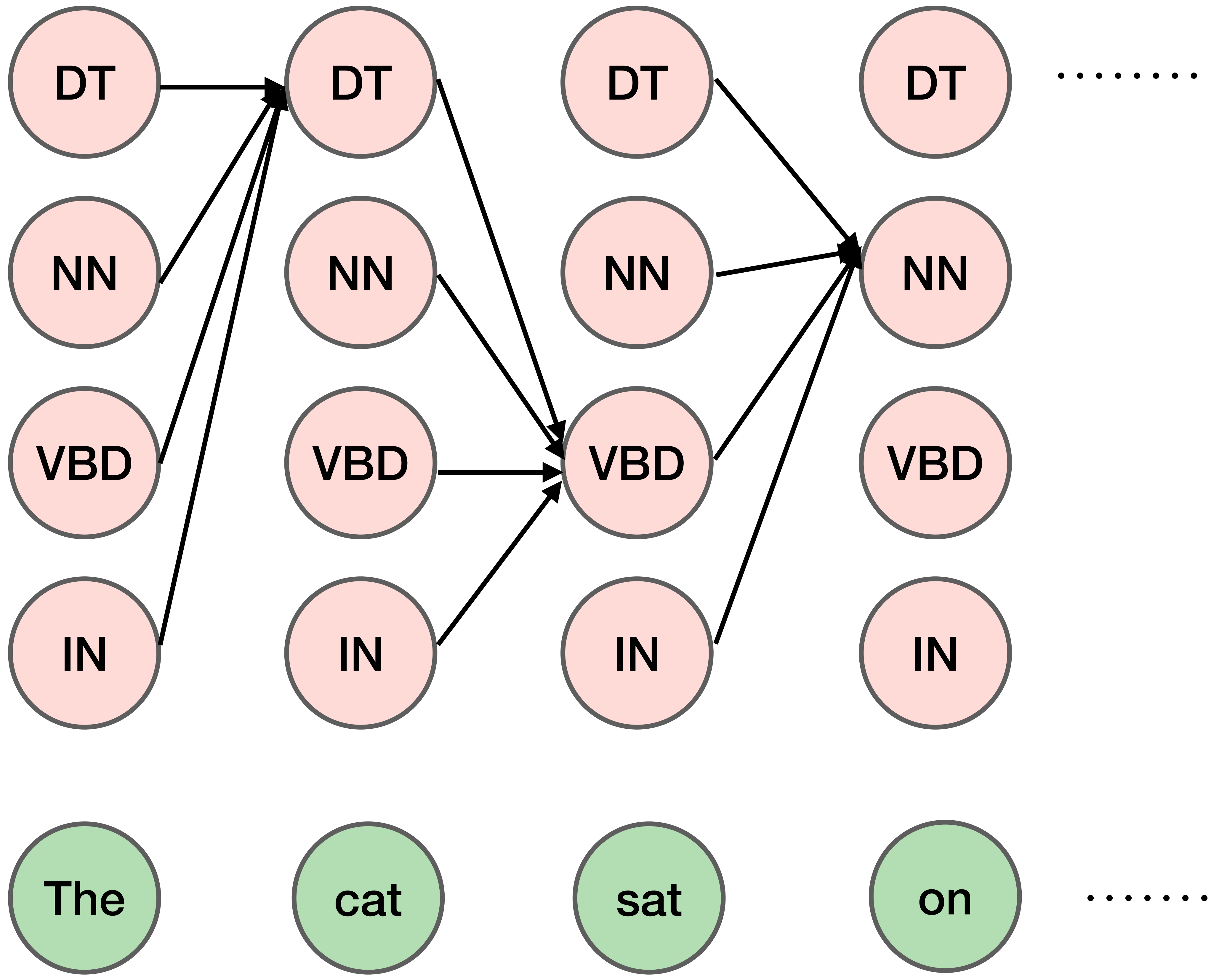
$$M[2,NN] = \max_k \{M[1,k]\ P(NN \mid k)\ P(\text{cat} \mid NN)\}$$

$$M[2,NN] = \max_k \{M[1,k] + \log P(NN \mid k) + \log P(\text{cat} \mid NN)\}$$

- In practice, we maximize sum of log probabilities (or minimize the sum of negative log probabilities) instead of maximize the product of probabilities

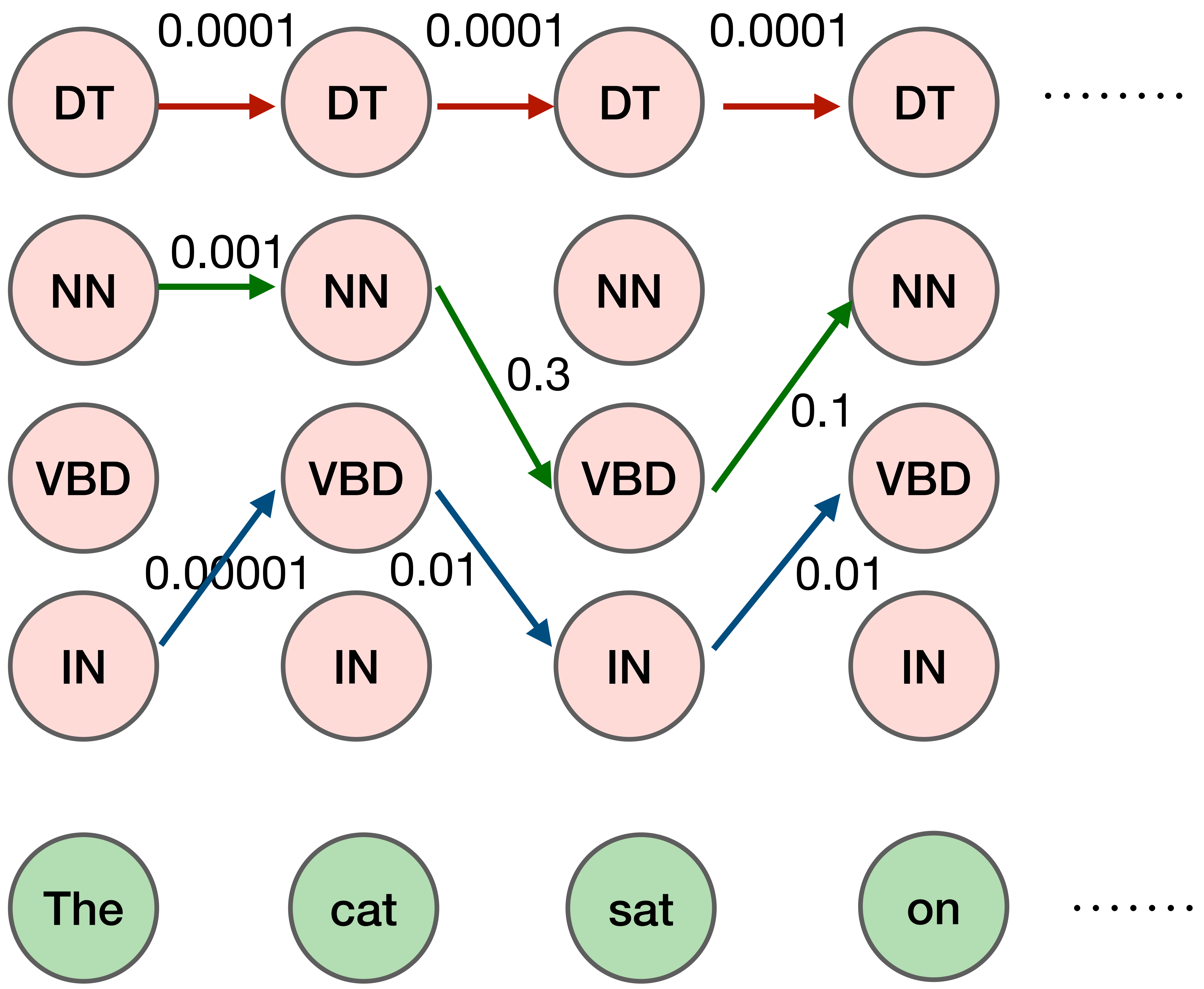
Beam search

- If K (number of possible hidden states) is too large, Viterbi is too expensive!



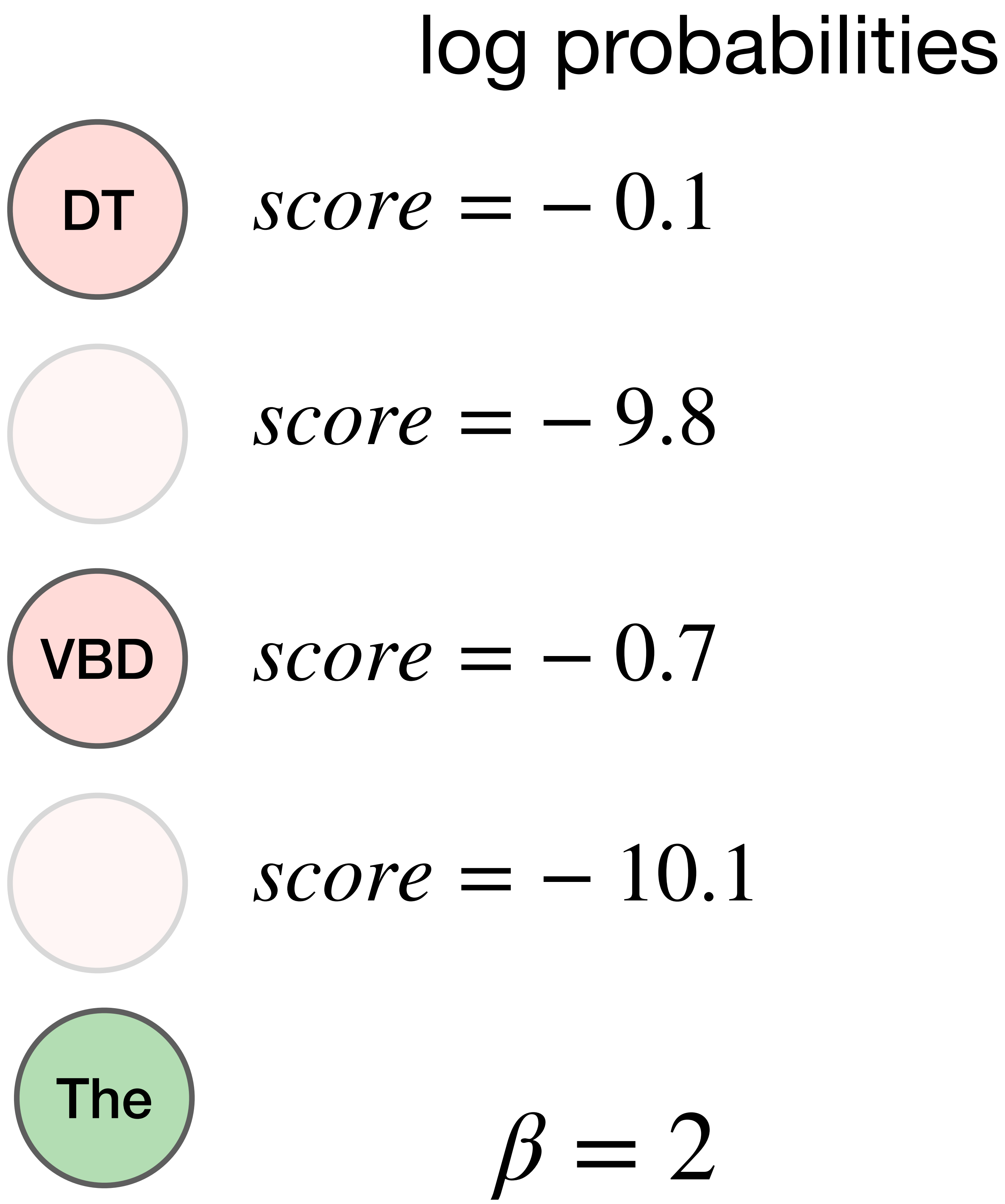
Beam search

- If K (number of possible hidden states) is too large, Viterbi is too expensive!
- **Observation:** Many paths have very low likelihood!



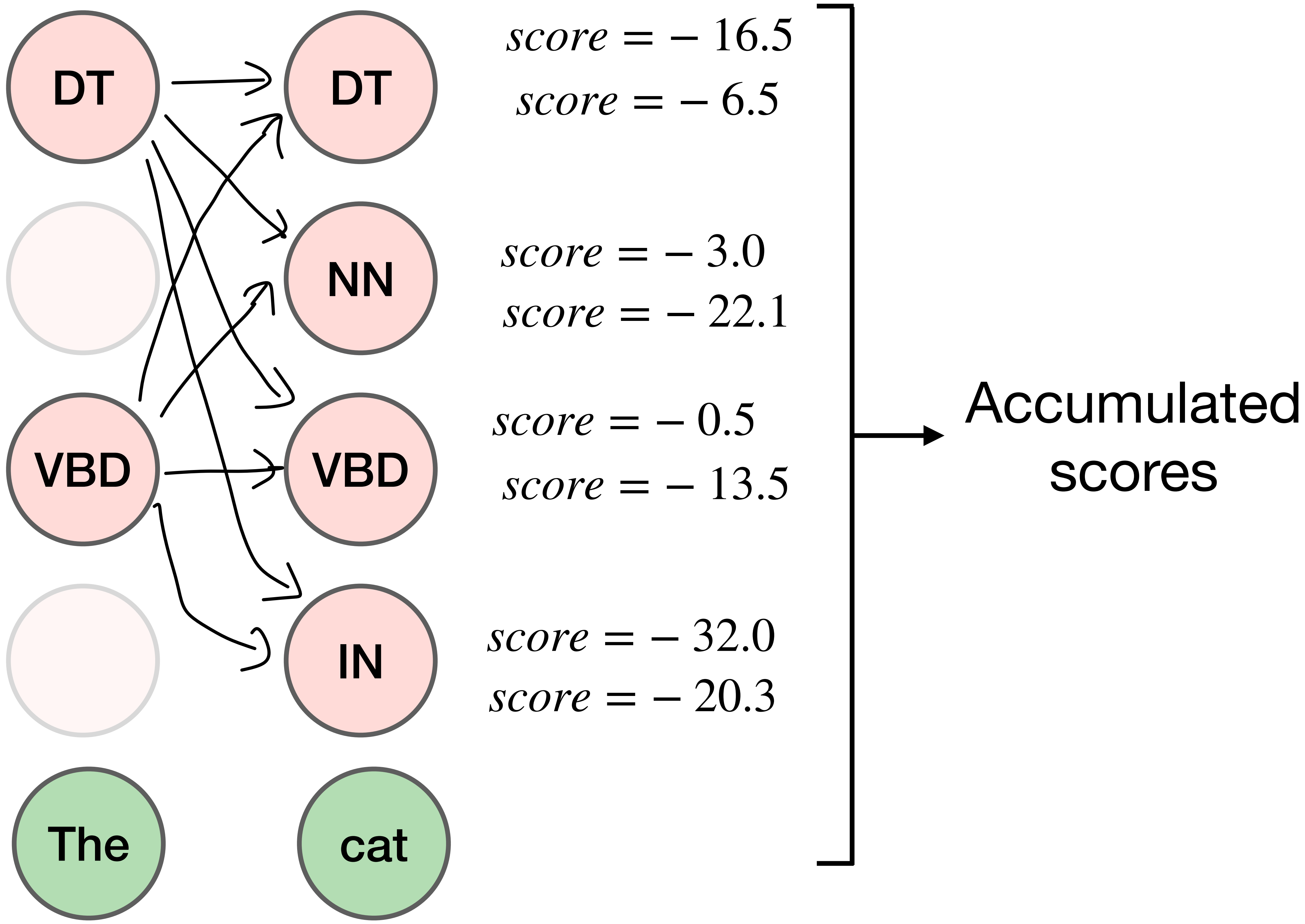
Beam search

- If K (number of possible hidden states) is too large, Viterbi is too expensive!
- **Observation:** Many paths have very low likelihood!
- Keep a fixed number of hypotheses at each point
    - Beam width = β



Beam search

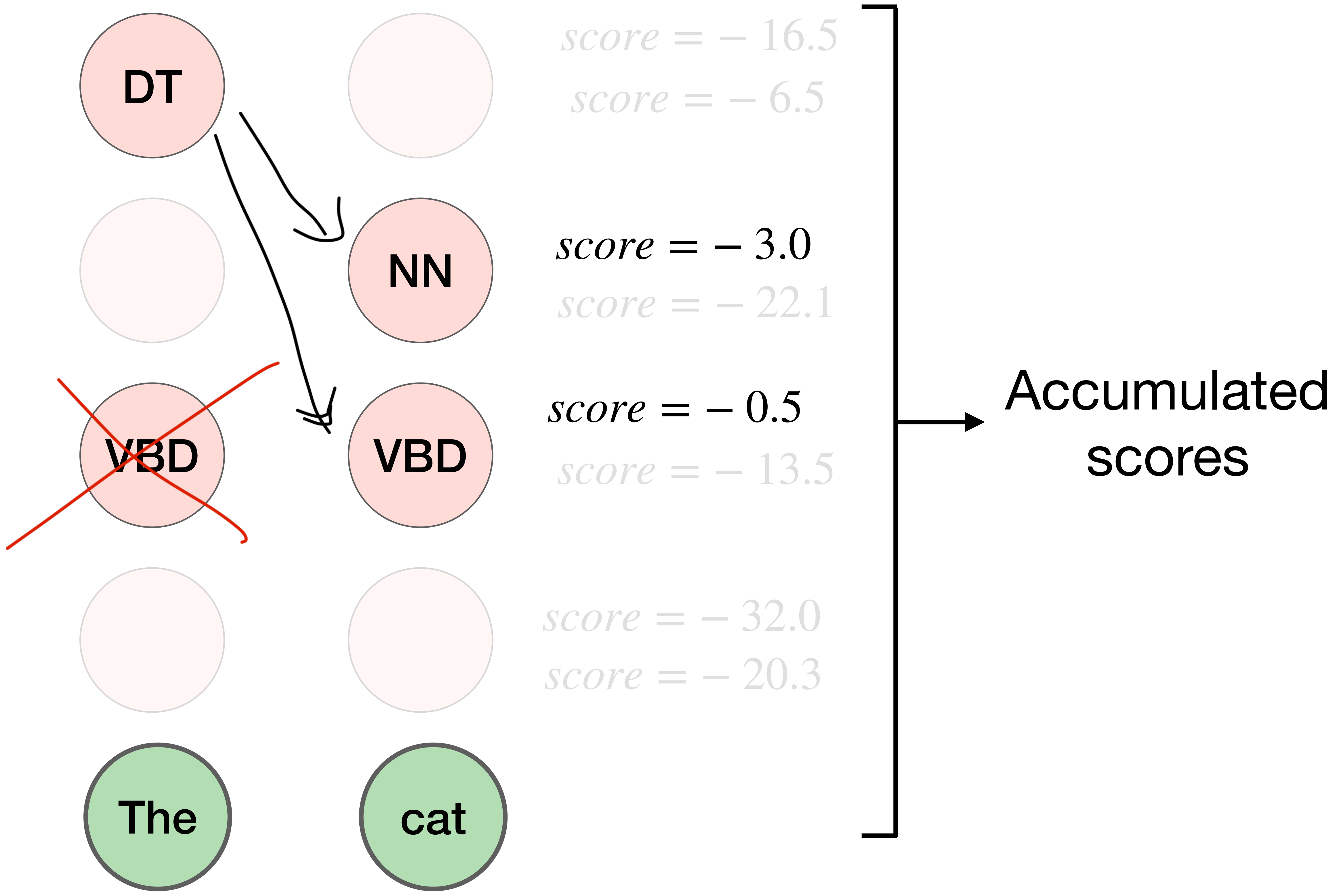
- If K (number of possible hidden states) is too large, Viterbi is too expensive!
- **Observation:** Many paths have very low likelihood!
- Keep a fixed number of hypotheses at each point
    - Beam width = β



Step 1: Expand all partial sequences in current beam

Beam search

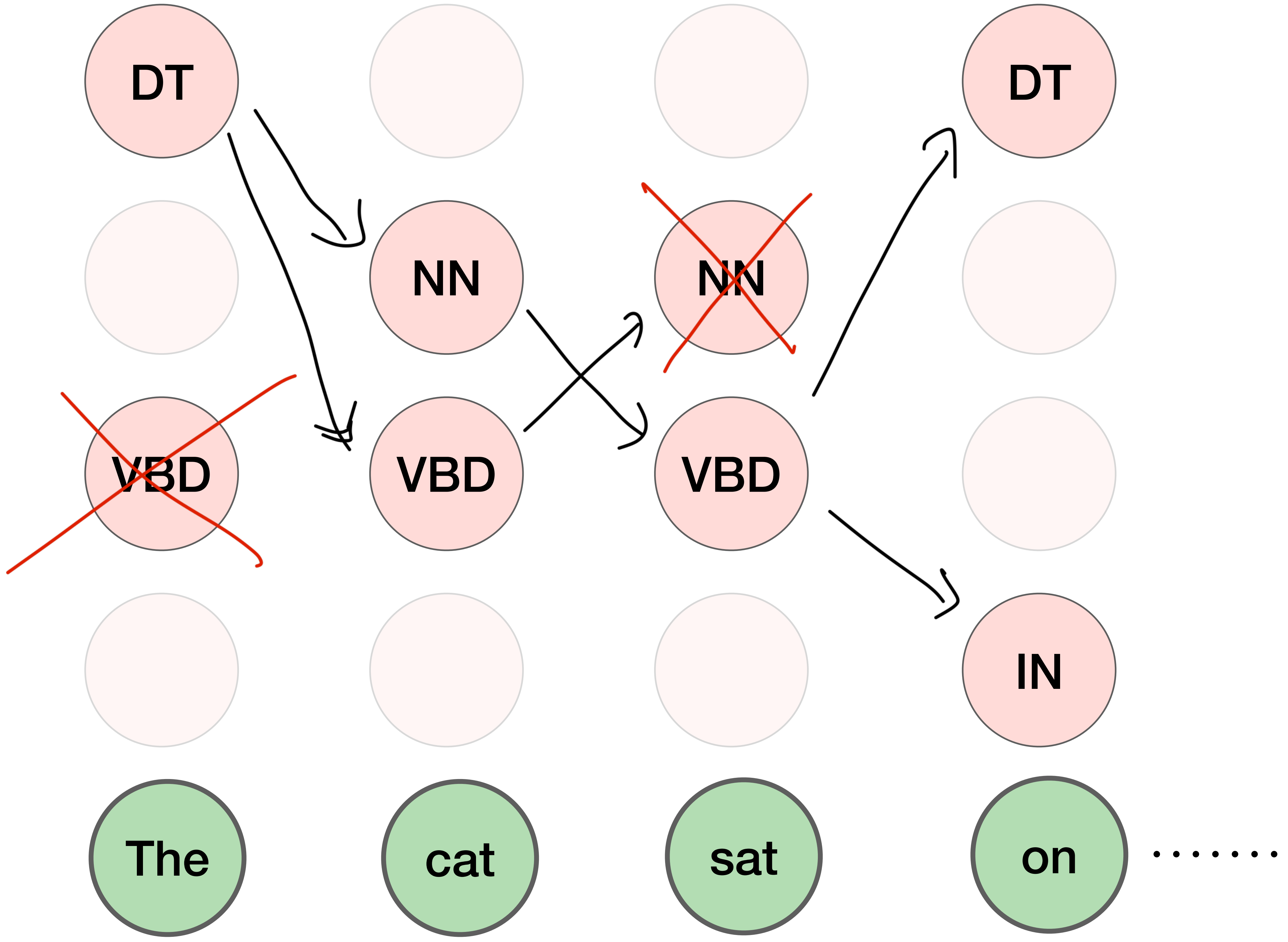
- If K (number of possible hidden states) is too large, Viterbi is too expensive!
- **Observation:** Many paths have very low likelihood!
- Keep a fixed number of hypotheses at each point
  - Beam width = β



Step 2: Prune back to top β scores (sort and select) … repeat!

Beam search

- If K (number of possible hidden states) is too large, Viterbi is too expensive!
- **Observation:** Many paths have very low likelihood!
- Keep a fixed number of hypotheses at each point
  - Beam width = β



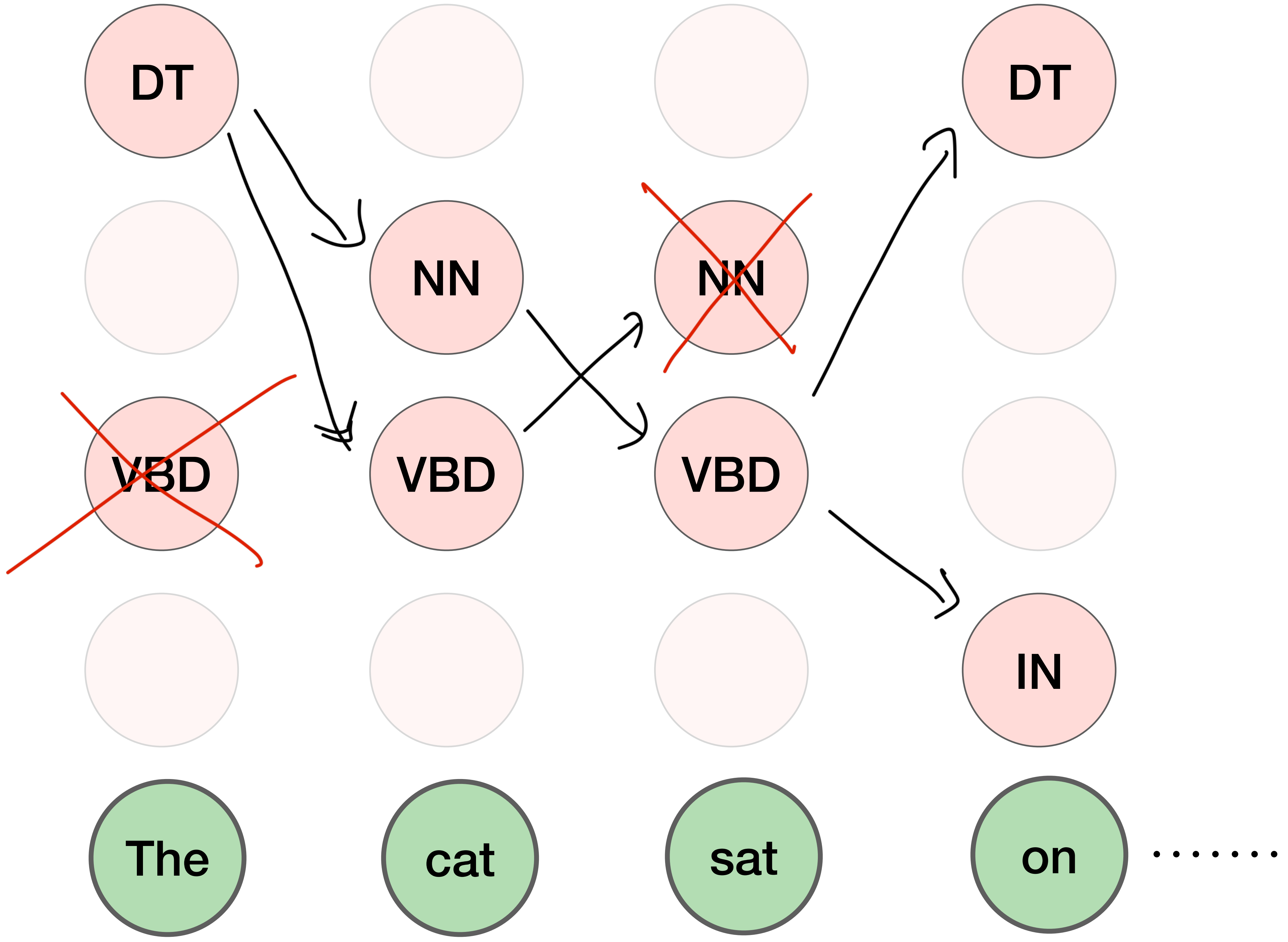
Pick $\max_k M[n,k]$ from within beam and backtrack

Beam search

What is the time complexity of Beam search? Assume n = number of timesteps, K = number of states, β = beam width

(A) $O(n\beta)$
(B) $O(nK\beta)$
(C) $O(n\beta^2)$
(D) $O(nK\beta^2)$

The answer is *(B)*: $O(nK\beta)$



Pick $\max_k M[n,k]$ from within beam and backtrack

Wrap up

- Hidden Markov models
- Viterbi algorithm
  - Use Markov assumption and dynamic programming to find optimal sequence of states
- Beam search
  - If number of states is too large, Viterbi is too expensive! Trade-off (some) accuracy for computational savings