



CMPT 413/713: Natural Language Processing

Sequence Models - HMMs

Spring 2024
2024-01-29

Adapted from slides from Danqi Chen and Karthik Narasimhan

Summary of topics

- So far
 - Language Modeling: $P(w_i | w_{1:i-1})$
 - Text classification: $P(c | d)$
 - with Naive Bayes, Logistic Regression and Neural Networks
 - Word embeddings: Representing w as a vector
- This week + next week
 - Sequence modeling
 - Transformers and contextual word-embeddings
- Later
 - Decoding and text generation
 - Structured prediction (parsing)
 - NLP Applications

Overview

- What is sequence modeling?
- Hidden markov models (HMM)
- Decoding algorithms: Greedy, Viterbi, Beam
- Maximum entropy markov models (MEMM)

Sequence Tagging (part-of-speech)

Input: sequence of words; Output: sequence of labels

Input	British	left	waffles	on	Falkland	Islands
Output1	N	N	V	P	N	N
Output2	N	V	N	P	N	N
:						

N Noun, e.g. islands

V Verb, e.g. leave, left

P Preposition, e.g. on

Other sequence tagging problems

BIO encoding

- Named Entity Recognition

B-PER I-PER O O O O B-ORG I-ORG
Michael Jordan is a professor at UC Berkeley.

- Shallow Phrase Chunking

B-NP I-NP B-VP B-PP B-NP I-NP B-PP B-NP I-NP
The cat sat on the mat under the sun

Sequence Tagging (phrase chunking)

[NP He] [VP reckons] [NP the current account deficit] [VP will narrow]

B-NP B-VP B-NP I-NP I-NP I-NP B-VP I-VP

PRP VBZ DT JJ NN NN MD VB

He reckons the current account deficit will Narrow

Example from <https://www.cnts.ua.ac.be/conll2000/chunking/>

Part of speech tags

What are POS tags?

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

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

British/**NNP** left/**NN** waffles/**NNS** on/**IN** Falkland/**NNP** Islands/**NNP**

The/**DT** old/**NN** man/**VB** the/**DT** boat/**NN**

Parts of Speech

- Different words have different functions
- **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



Penn Tree Bank tagset

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

[45 tags]

Figure 8.1 Penn Treebank part-of-speech tags (including punctuation).

(Marcus et al., 1993)

Other corpora: Brown, WSJ, Switchboard

Part of Speech Tagging

- Disambiguation task: each word might have different senses/functions
- The/DT **man/NN** bought/VBD a/DT boat/NN
- The/DT old/NN **man/VB** the/DT boat/NN

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%)

Figure 8.2 Tag ambiguity for word types in Brown and WSJ, using Treebank-3 (45-tag) tagging. Punctuation were treated as words, and words were kept in their original case.

Part of Speech Tagging

- Disambiguation task: each word might have different senses/functions
- The/DT **man/NN** bought/VBD a/DT boat/NN
- The/DT old/NN **man/VB** the/DT boat/NN

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!

A simple baseline

- Many words might be easy to disambiguate
- **Most frequent class:** Assign each token (word) to the class it occurred most in the training set. (e.g. man/NN)
- Accurately tags **92.34%** of word tokens on Wall Street Journal (WSJ)!
- State of the art ~97-98%
- Average English sentence ~ 14 words
 - Sentence level accuracies: $0.92^{14} = \mathbf{31\%}$ vs $0.97^{14} = \mathbf{65\%}$ vs $0.98^{14} = \mathbf{75\%}$

The/DT **old/JJ** **man/NN** the/DT boat/NN

- POS tagging not solved yet!

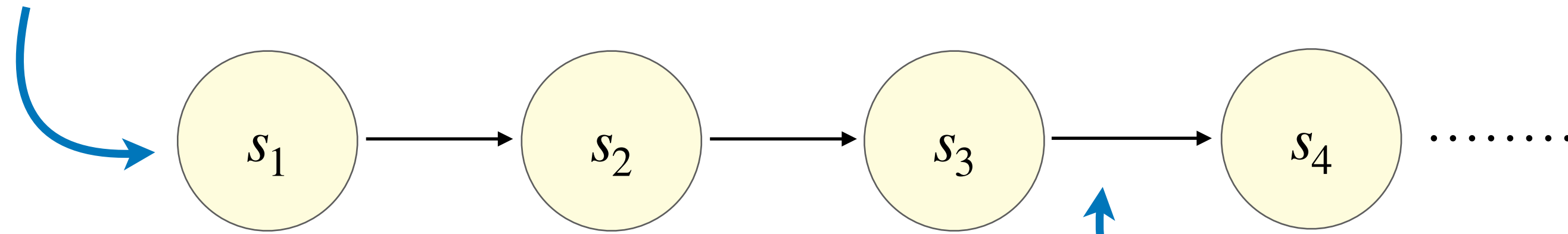
Hidden Markov Models

Some observations

- The function (or POS) of a word depends on its context
 - The/DT **old/NN** **man/VB** the/DT boat/NN
 - The/DT **old/JJ** **man/NN** bought/VBD the/DT boat/NN
- Certain POS combinations are extremely unlikely
 - $\langle JJ, DT \rangle$ or $\langle DT, IN \rangle$
- Better to make decisions on entire sequences instead of individual words (Sequence modeling!)

Markov chains

$\pi(s_1)$: Initial distribution

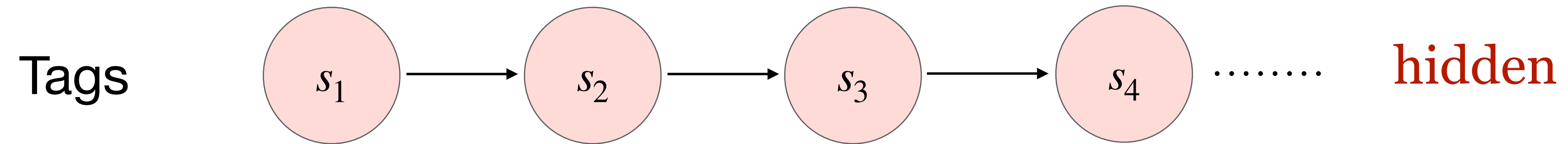


$P(s_t | s_{t-1})$: Transition probability

- Model probabilities of sequences of variables
- Each state can take one of K values ($\{1, 2, \dots, K\}$ for simplicity)
- Markov assumption: $P(s_t | s_{<t}) \approx P(s_t | s_{t-1})$

Where have we seen this before?

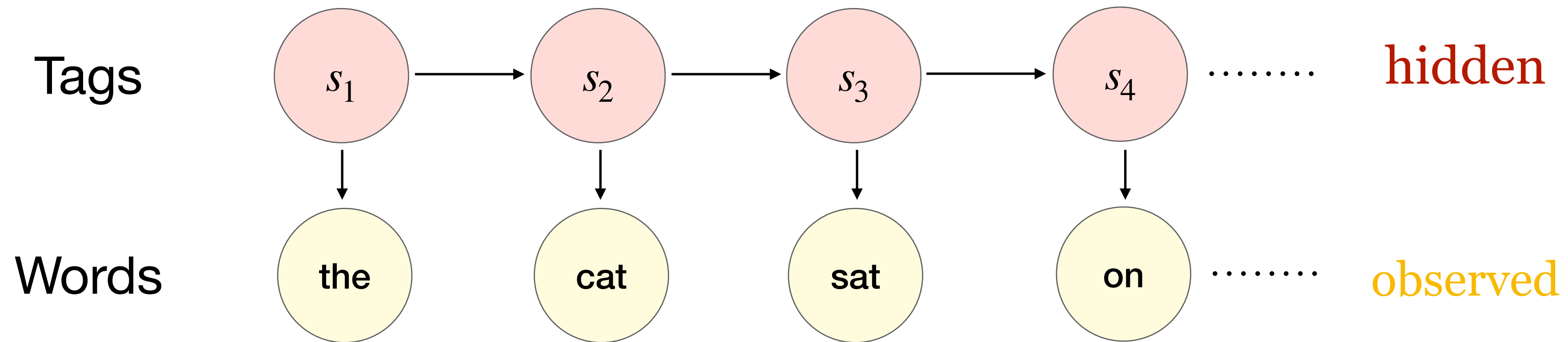
Markov chains



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

- We don't observe POS tags at test time

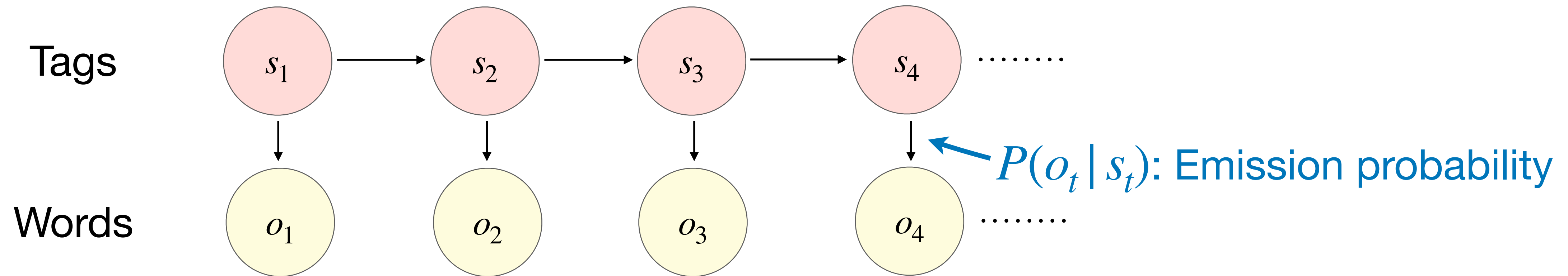
Hidden Markov Model (HMM)



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

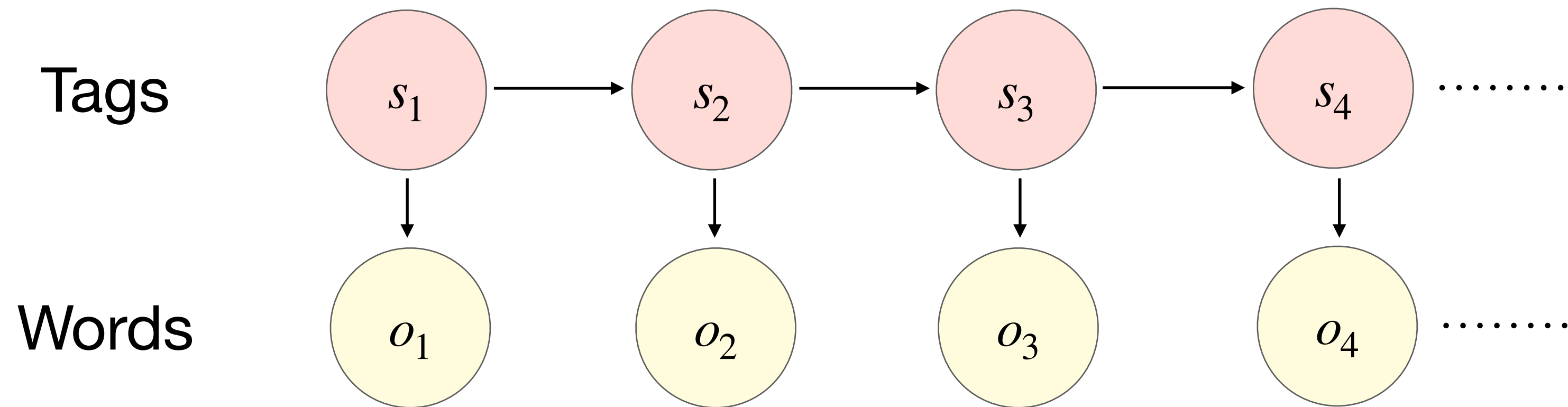
- We don't observe POS tags at test time
- But we do observe the words!
- HMM allows us to *jointly reason* over both **hidden** and **observed** events.

Components of an HMM



1. Set of states $S = \{1, 2, \dots, K\}$ and observations O
2. Initial state probability distribution $\pi(s_1)$
3. Transition probabilities $P(s_{t+1} | s_t)$
4. Emission probabilities $P(o_t | s_t)$

Assumptions



1. Markov assumption:

$$P(s_{t+1} | s_1, \dots, s_t) = P(s_{t+1} | s_t)$$

Transition
Probabilities

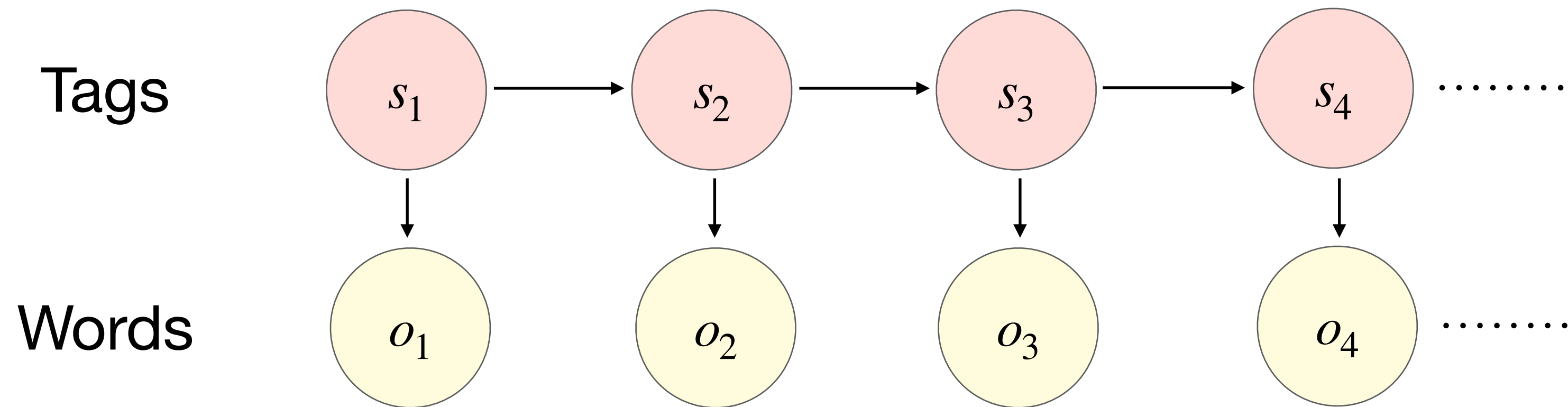
2. Output independence:

$$P(o_t | s_1, \dots, s_t) = P(o_t | s_t)$$

Emission
Probabilities

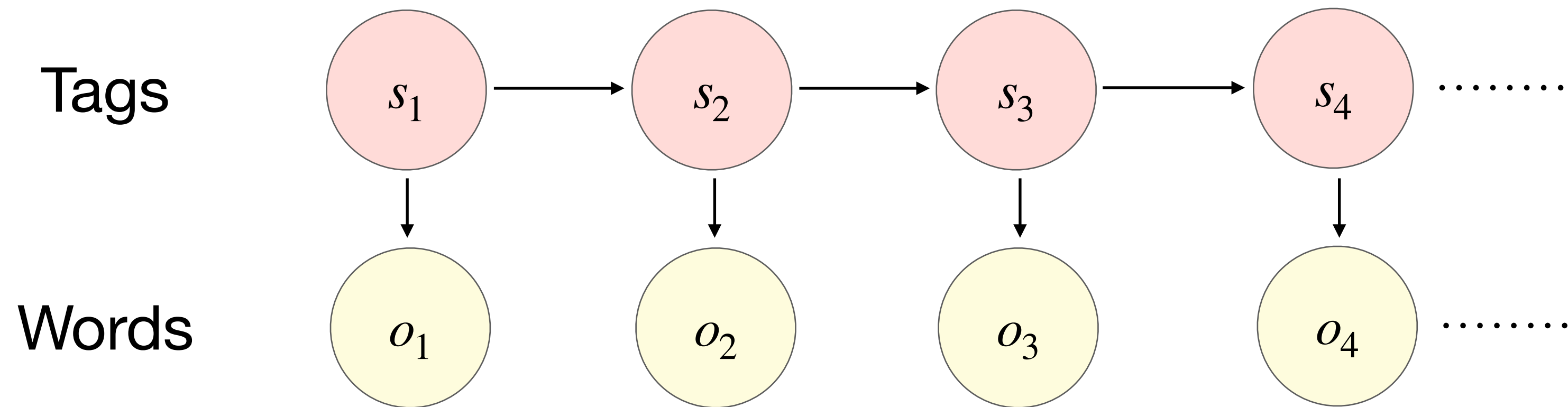
Which is a stronger assumption?

What we want to be able to do?



- Compute the probability (likelihood) of a sequence
- Estimate the model parameters (train the model)
- Use the model to generate new sequences (decoding)

Sequence likelihood



$$\begin{aligned} P(S, O) &= P(s_1, s_2, \dots, s_n, o_1, o_2, \dots, o_n) \\ &= \pi(s_1) P(o_1 | s_1) \prod_{t=2}^n P(s_t, o_t | s_{t-1}) \\ &= \pi(s_1) P(o_1 | s_1) \prod_{t=2}^n P(o_t | s_t) P(s_t | s_{t-1}) \end{aligned}$$

Example: POS tagging

$\pi(DT) = 0.8$ the/DT cat/NN sat/VBD on/IN the/DT mat/NN

		s_{t+1}					o_t						...
		DT	NN	IN	VBD		the	cat	sat	on	mat		
s_t	DT	0.05	0.8	0.05	0.1	DT	0.5	0	0	0	0		
	NN	0.05	0.2	0.15	0.6	NN	0.01	0.2	0.01	0.01	0.2		
	IN	0.5	0.2	0.05	0.25	IN	0	0	0	0.4	0		
	VBD	0.3	0.3	0.3	0.1	VBD	0	0.01	0.1	0.01	0.01		

P(the|DT, cat|NN, sat|VBD, on|IN, the|DT, mat|NN) = ??

Example: POS tagging

$$\pi(DT) = 0.8$$

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

s_{t+1}

o_t

s_t

	DT	NN	IN	VBD
DT	0.05	0.8	0.05	0.1
NN	0.05	0.2	0.15	0.6
IN	0.5	0.2	0.05	0.25
VBD	0.3	0.3	0.3	0.1

	the	cat	sat	on	mat
DT	0.5	0	0	0	0
NN	0.01	0.2	0.01	0.01	0.2
IN	0	0	0	0.4	0
VBD	0	0.01	0.1	0.01	0.01

Where did these numbers come from?

Learned from data!

$P(\text{the}|\text{DT}, \text{cat}|\text{NN}, \text{sat}|\text{VBD}, \text{on}|\text{IN}, \text{the}|\text{DT}, \text{mat}|\text{NN}) =$

$$= \overset{0.8}{\pi(\text{DT})} \overset{0.5}{P(\text{the}|\text{DT})} \overset{0.8}{P(\text{NN}|\text{DT})} \overset{0.2}{P(\text{cat}|\text{NN})} \overset{0.6}{P(\text{VBD}|\text{NN})} \overset{0.1}{P(\text{sat}|\text{VBD})} \dots$$

$$= 1.84 * 10^{-5}$$

Learning for HMMs

Learning from fully observed data

Fully labeled! All tags are
known during training

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** Hurricane/**NNP** Hugo/**NNP** victims/**NNS** ,/, and/**CC** sending/**VBG** them/**PRP** to/**TO** San/**NNP** Francisco/**NNP** instead/**RB** ./.

Learning from fully observed data

Training set:

1 Pierre/**NNP** Vinken/**NNP** ,/, 61/**CD** year
join/**VB** the/**DT** board/**NN** as/**IN** a/**DT** no
Nov./**NNP** 29/**CD** ./.

2 Mr./**NNP** Vinken/**NNP** is/**VBZ** chairman
N.V./**NNP** ,/, the/**DT** Dutch/**NNP** publish

3 Rudolph/**NNP** Agnew/**NNP** ,/, 55/**CD** ye
chairman/**NN** of/**IN** Consolidated/**NNP** Go
/, was/**VBD** named/**VBN** a/**DT** nonexecut
this/**DT** British/**JJ** industrial/**JJ** conglomer

...

38,219 It/**PRP** is/**VBZ** also/**RB** pulling/**VB**
of/**IN** Puerto/**NNP** Rico/**NNP** ,/, who/**WP**
Hurricane/**NNP** Hugo/**NNP** victims/**NNS** ,/
them/**PRP** to/**TO** San/**NNP** Francisco/**NN**

Easy!

Maximum likelihood
estimate:

$$P(s_i | s_j) = \frac{C(s_j, s_i)}{C(s_j)}$$

$$P(o | s) = \frac{C(s, o)}{C(s)}$$

Estimating probabilities

$$\pi(DT) = \frac{2}{2}$$

s_{t+1}

o_t

s_t

	DT	NN	IN	VBD
DT				
NN				
IN				
VBD				

	the	cat	sat	on	mat	a	cleaned	chair	man
DT									
NN									
IN									
VBD									

Training corpus:

a/DT man/NN cleaned/VBD the/DT mat/NN
the/DT cat/NN sat/VBD on/IN the/DT chair/NN

Estimating probabilities

$$\pi(DT) = \frac{2}{2}$$

s_{t+1}

o_t

s_t

	DT	NN	IN	VBD
DT				
NN				
IN				
VBD				

	the	cat	sat	on	mat	a	cleaned	chair	man
DT	3/4					1/4			
NN									
IN									
VBD									

Training corpus:

a/DT man/NN cleaned/VBD the/DT mat/NN
the/DT cat/NN sat/VBD on/IN the/DT chair/NN

Estimating probabilities

$$\pi(DT) = \frac{2}{2}$$

s_{t+1}

o_t

s_t

	DT	NN	IN	VBD
DT				
NN				
IN				
VBD				

	the	cat	sat	on	mat	a	cleaned	chair	man
DT	3/4					1/4			
NN		1/4			1/4			1/4	1/4
IN				1/1					
VBD			1/2				1/2		

Training corpus:

a/DT man/NN cleaned/VBD the/DT mat/NN
the/DT cat/NN sat/VBD on/IN the/DT chair/NN

Estimating probabilities

$$\pi(DT) = \frac{2}{2}$$

s_{t+1}

o_t

s_t

	DT	NN	IN	VBD
DT				
NN				
IN				
VBD	1/2		1/2	

	the	cat	sat	on	mat	a	cleaned	chair	man
DT	3/4					1/4			
NN		1/4			1/4			1/4	1/4
IN				1/1					
VBD			1/2				1/2		

Training corpus:

a/DT man/NN cleaned/VBD the/DT mat/NN
the/DT cat/NN sat/VBD on/IN the/DT chair/NN

Estimating probabilities

$$\pi(DT) = \frac{2}{2}$$

s_{t+1}

s_t

	DT	NN	IN	VBD	EOS
DT		4/4			
NN				2/4	2/4
IN	1/1				
VBD	1/2		1/2		

o_t

	the	cat	sat	on	mat	a	cleaned	chair	man
DT	3/4					1/4			
NN		1/4			1/4			1/4	1/4
IN				1/1					
VBD			1/2				1/2		

Training corpus:

a/DT man/NN cleaned/VBD the/DT mat/NN
the/DT cat/NN sat/VBD on/IN the/DT chair/NN

Learning from partially observable data (unsupervised learning)

No labels (or partial labels).

Still want to estimate parameters to maximize likelihood of training data.

Parameters: $\theta = \{P(s_i | s_j), P(o | s)\}$

Guaranteed to iteratively
improve likelihood

$$L(\theta_t) \geq L(\theta_{t-1})$$

EM for HMMs is also known as

Baum-Welch

EM: Expectation-Maximization algorithm

Initialize parameters to some random values

E-Step: Compute **expected** counts \bar{C} using current parameters

M-Step: Take expected counts use it to re-estimate parameters that **maximizes** the likelihood

$$P(s_i | s_j) = \frac{\bar{C}(s_j, s_i)}{\bar{C}(s_j)}, \quad P(o | s) = \frac{\bar{C}(s, o)}{\bar{C}(s)}$$

Iterate until convergence.

Decoding for HMMs

(finding the best sequence)

Example: POS tagging

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

$\pi(DT) = 0.8$

S_{t+1}

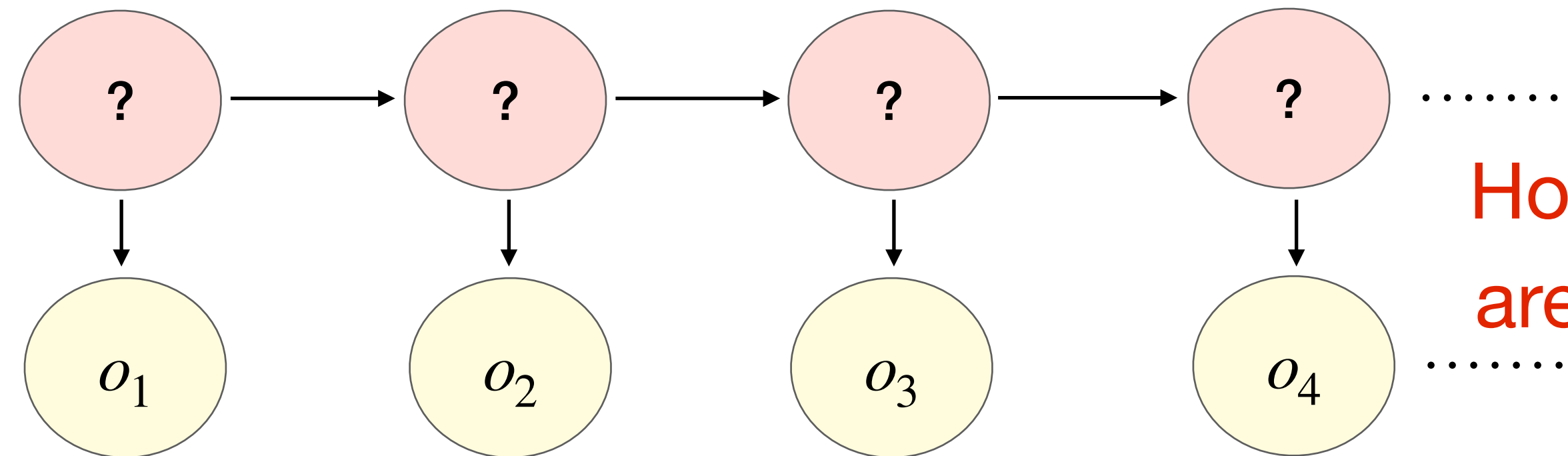
O_t

S_t		DT	NN	IN	VBD
	DT	0.05	0.8	0.05	0.1
	NN	0.05	0.2	0.15	0.6
	IN	0.5	0.2	0.05	0.25
	VBD	0.3	0.3	0.3	0.1

	the	cat	sat	on	mat
DT	0.5	0	0	0	0
NN	0.01	0.2	0.01	0.01	0.2
IN	0	0	0	0.4	0
VBD	0	0.01	0.1	0.01	0.01

How to find best sequence?

Decoding with HMMs



How many sequence of states
are there for $s_t \in \{1, 2, \dots, K\}$?

K^n

- **Task:** Find the most probable sequence of states

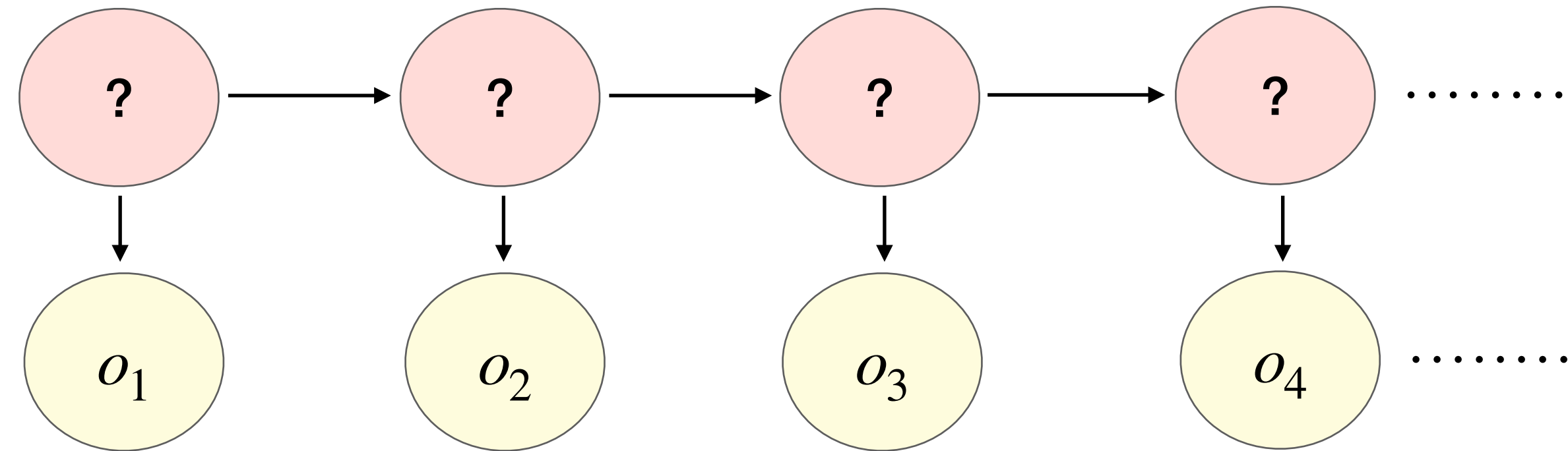
$\langle s_1, s_2, \dots, s_n \rangle$ given the observations $\langle o_1, o_2, \dots, o_n \rangle$

What is the best sequence of tags for the observed sequence: the cat sat on the mat

$$\hat{S} = \arg \max_S P(S|O) = \arg \max_S \frac{P(S)P(O|S)}{P(O)} \quad \text{Bayes Rule}$$

$$\hat{S} = \arg \max_S P(S)P(O|S)$$

Decoding with HMMs



- **Task:** Find the most probable sequence of states

$\langle s_1, s_2, \dots, s_n \rangle$ given the observations $\langle o_1, o_2, \dots, o_n \rangle$

$$\pi(\langle \text{SOS} \rangle) = 1$$

$$\hat{S} = \arg \max_S P(S) P(O|S)$$

$$P(s_1|s_0) = P(s_1 | \langle \text{SOS} \rangle)$$

or

$$P(s_1|s_0) = \pi(s_1)$$

$$= \arg \max_S \prod_{t=1}^n \underbrace{P(o_t | s_t)}_{\text{Emission Probabilities}} \underbrace{P(s_t, | s_{t-1})}_{\text{Transition Probabilities}}$$

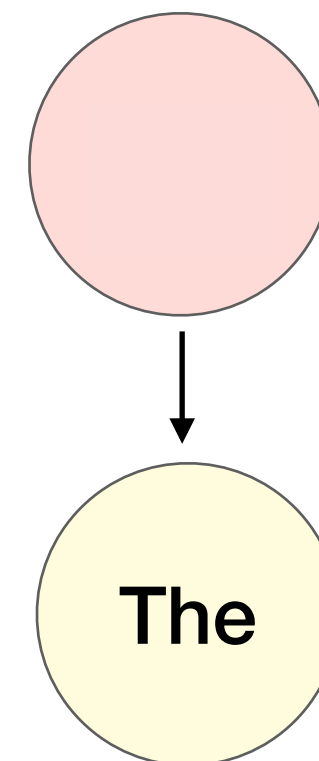
Greedy decoding

$$s$$

	DT	NN	IN	VBD
π	0.8	0.1	0.05	0.05

$$s_{t+1}$$

	DT	NN	IN	VBD
DT	0.05	0.8	0.05	0.1
NN	0.05	0.2	0.15	0.6
IN	0.5	0.2	0.05	0.25
VBD	0.3	0.3	0.3	0.1



$$\arg \max_s P(\text{The}|s)\pi(s_1 = s) = \text{DT}$$

$$o_t$$

	the	cat	sat	on	mat
DT	0.5	0	0	0	0
NN	0.01	0.2	0.01	0.01	0.2
IN	0	0	0	0.4	0
VBD	0	0.01	0.1	0.01	0.01

$$\hat{S} = \arg \max_S P(S)P(O|S)$$

$$= \arg \max_S \prod_{t=1}^n \underbrace{P(o_t|s_t)}_{\text{Emission}} \underbrace{P(s_t, |s_{t-1})}_{\text{Transition}}$$

Greedy decoding

S

	DT	NN	IN	VBD
π	0.8	0.1	0.05	0.05

S_{t+1}

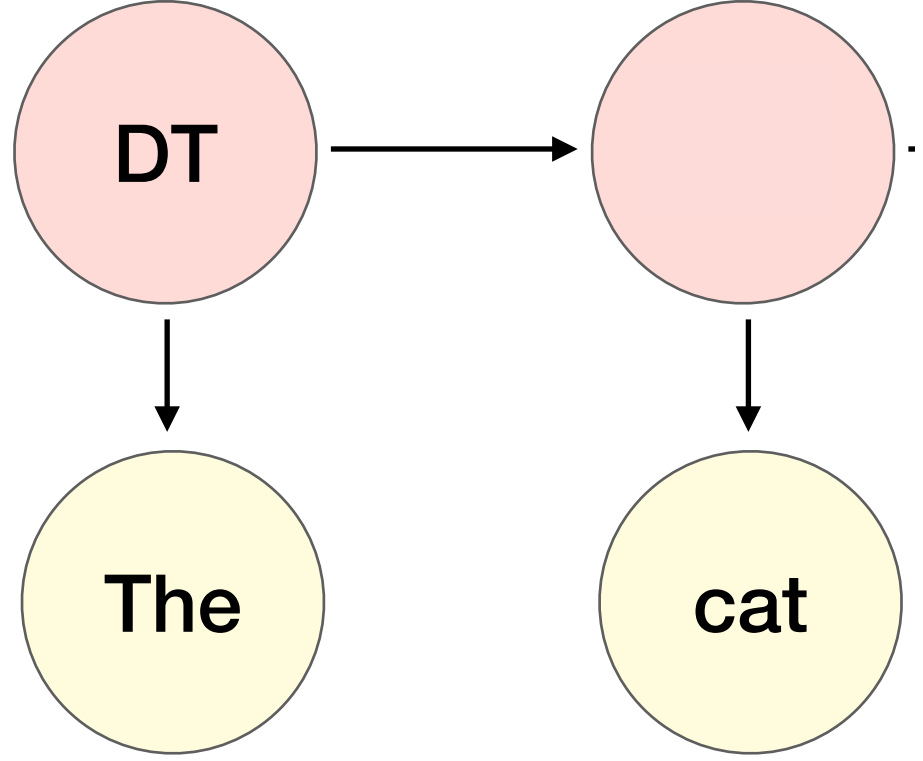
	DT	NN	IN	VBD
DT	0.05	0.8	0.05	0.1
NN	0.05	0.2	0.15	0.6
IN	0.5	0.2	0.05	0.25
VBD	0.3	0.3	0.3	0.1

S_t

O_t

	the	cat	sat	on	mat
DT	0.5	0	0	0	0
NN	0.01	0.2	0.01	0.01	0.2
IN	0	0	0	0.4	0
VBD	0	0.01	0.1	0.01	0.01

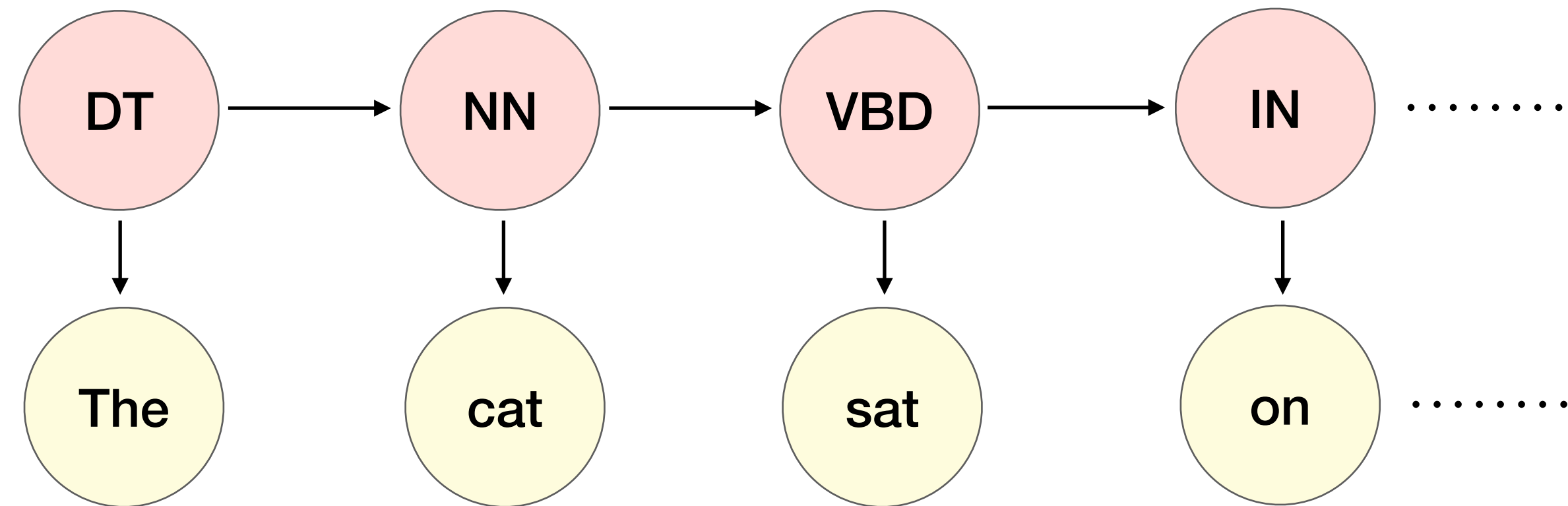
S_t



$$\arg \max_s P(\text{cat}|s)P(s_2 = s|DT) = \text{NN}$$

$$\begin{aligned} \hat{S} &= \arg \max_S P(S)P(O|S) \\ &= \arg \max_S \prod_{t=1}^n \underbrace{P(o_t|s_t)}_{\text{Emission}} \underbrace{P(s_t, |s_{t-1})}_{\text{Transition}} \end{aligned}$$

Greedy decoding



$$\forall t, \hat{s}_t = \arg \max_s P(o_t | s_t) P(s | \hat{s}_{t-1})$$

- Not guaranteed to be optimal!
- Local decisions
- Fast! $O(K \times n)$

Viterbi decoding

- Use **dynamic programming**!
- Probability lattice, $M[T, K]$
- T : Number of time steps
- K : Number of states

$M[T, K]$

T

	the	cat	sat	on	the	mat
DT						
NN						
IN						
VBD						

K

- $M[i, j]$: Most **probable** sequence of states ending with state **j** at time **i**

Viterbi decoding

$$S$$

	DT	NN	IN	VBD
π	0.8	0.1	0.05	0.05

$$S_{t+1}$$

	DT	NN	IN	VBD
DT	0.05	0.8	0.05	0.1
NN	0.05	0.2	0.15	0.6
IN	0.5	0.2	0.05	0.25
VBD	0.3	0.3	0.3	0.1

$$O_t$$

	the	cat	sat	on	mat
DT	0.5	0	0	0	0
NN	0.01	0.2	0.01	0.01	0.2
IN	0	0	0	0.4	0
VBD	0	0.01	0.1	0.01	0.01

DT

NN

IN

VBD

the

$$M[1,DT] = \pi(DT) P(\text{the} | DT) = 0.8 \times 0.5 = 0.4$$

$$M[1,NN] = \pi(NN) P(\text{the} | NN) = 0.1 \times 0.01 = 0.001$$

$$M[1,IN] = \pi(IN) P(\text{the} | IN) = 0.05 \times 0 = 0$$

$$M[1,VBD] = \pi(VBD) P(\text{the} | VBD) = 0.05 \times 0 = 0$$

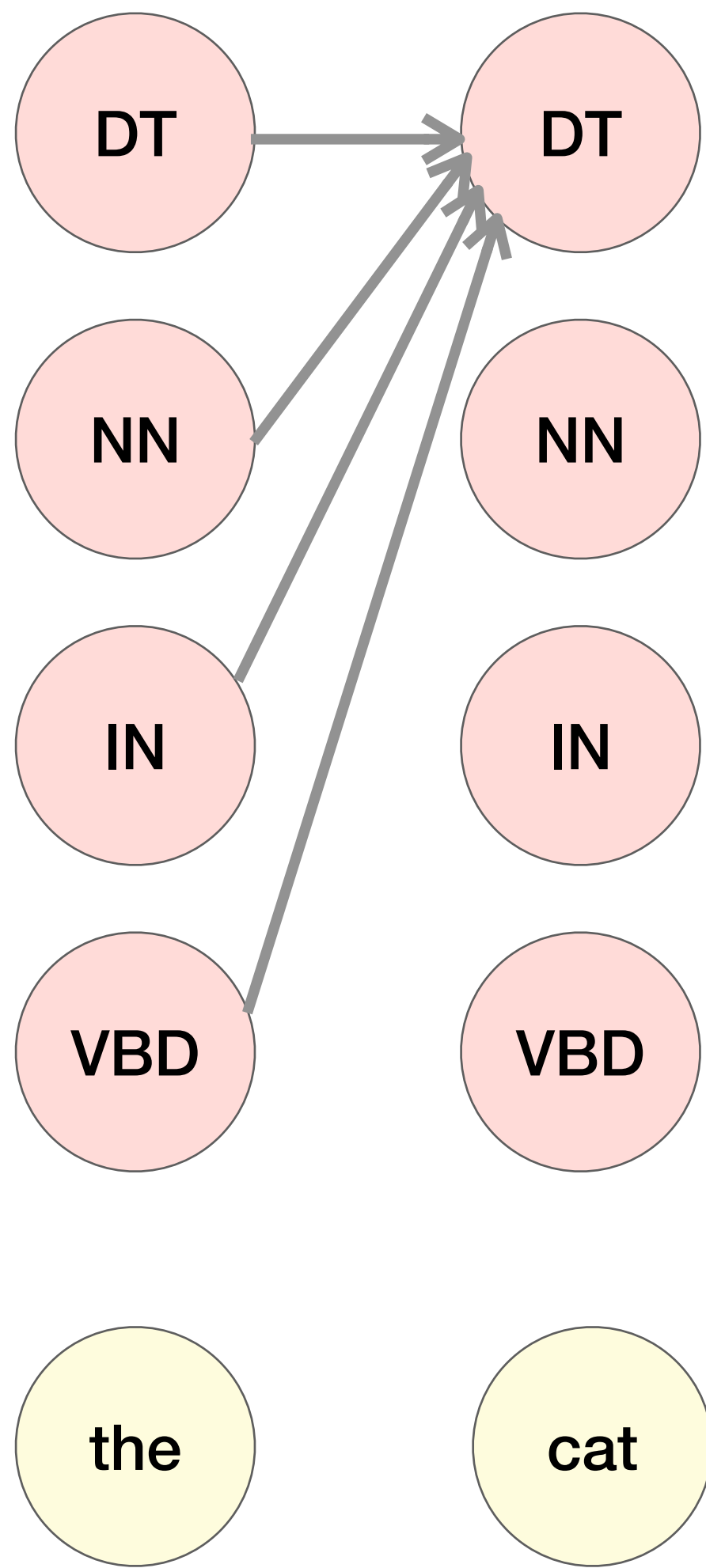
Forward

$$M[T, K]$$

	the	cat	sat	on	the	mat
DT	0.4					
NN	0.001					
IN	0					
VBD	0					

Viterbi decoding

		s_{t+1}				
		DT	NN	IN	VBD	
s_t	DT	0.05	0.8	0.05	0.1	
	NN	0.05	0.2	0.15	0.6	
	IN	0.5	0.2	0.05	0.25	
	VBD	0.3	0.3	0.3	0.1	
		o_t				
		the	cat	sat	on	mat
s_t	DT	0.5	0	0	0	0
	NN	0.01	0.2	0.01	0.01	0.2
	IN	0	0	0	0.4	0
	VBD	0	0.01	0.1	0.01	0.01

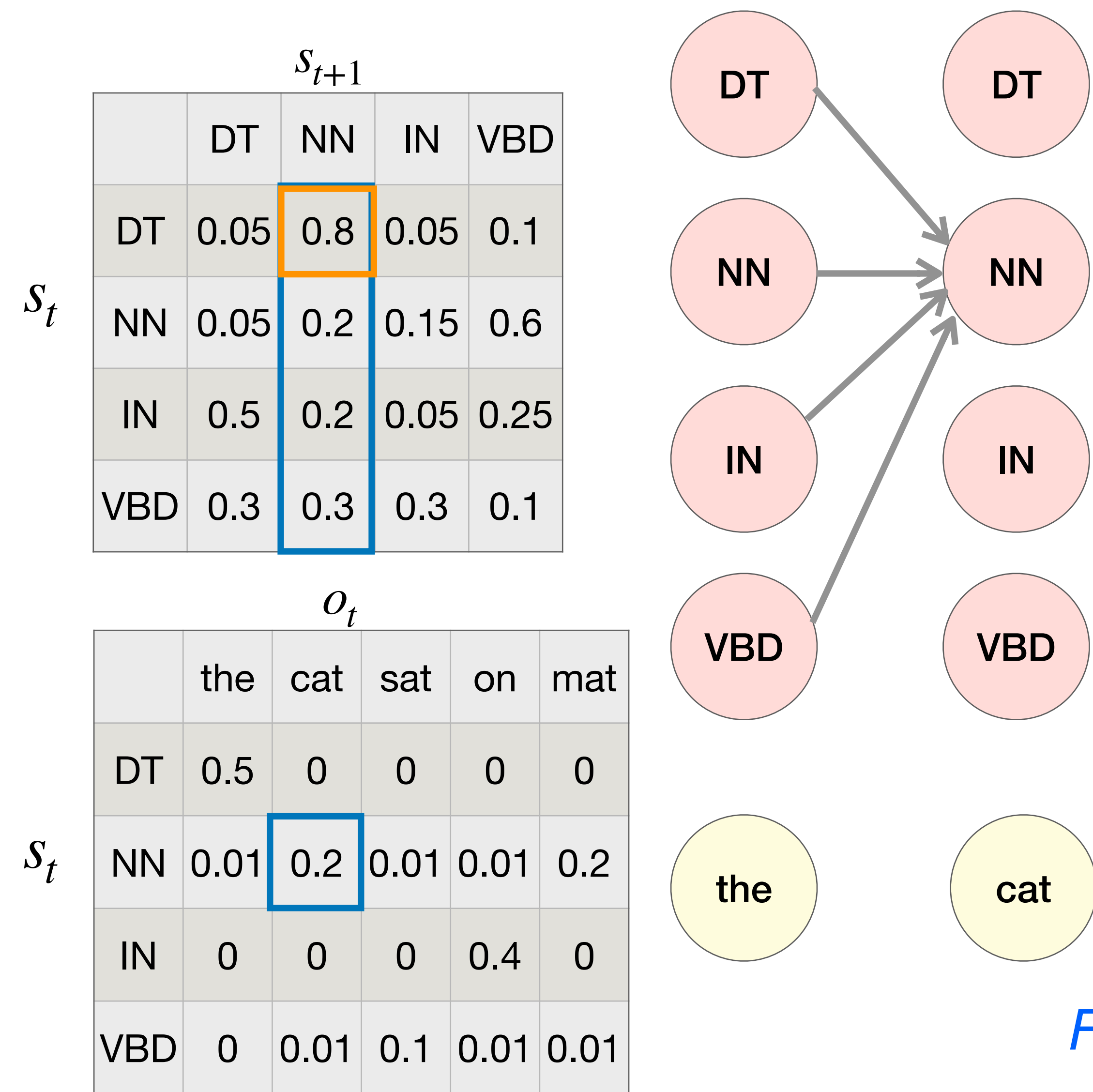


$$M[2,DT] = \max_k M[1,k] P(DT|k) P(\text{cat}|DT) = 0$$

	the	cat	sat	on	the	mat
DT	0.4	0				
NN	0.001					
IN	0					
VBD	0					
		$M[1,K]$				

Forward

Viterbi decoding



$$M[2, NN] = \max_k M[1, k] P(NN | k) P(\text{cat} | NN) = 0.064$$

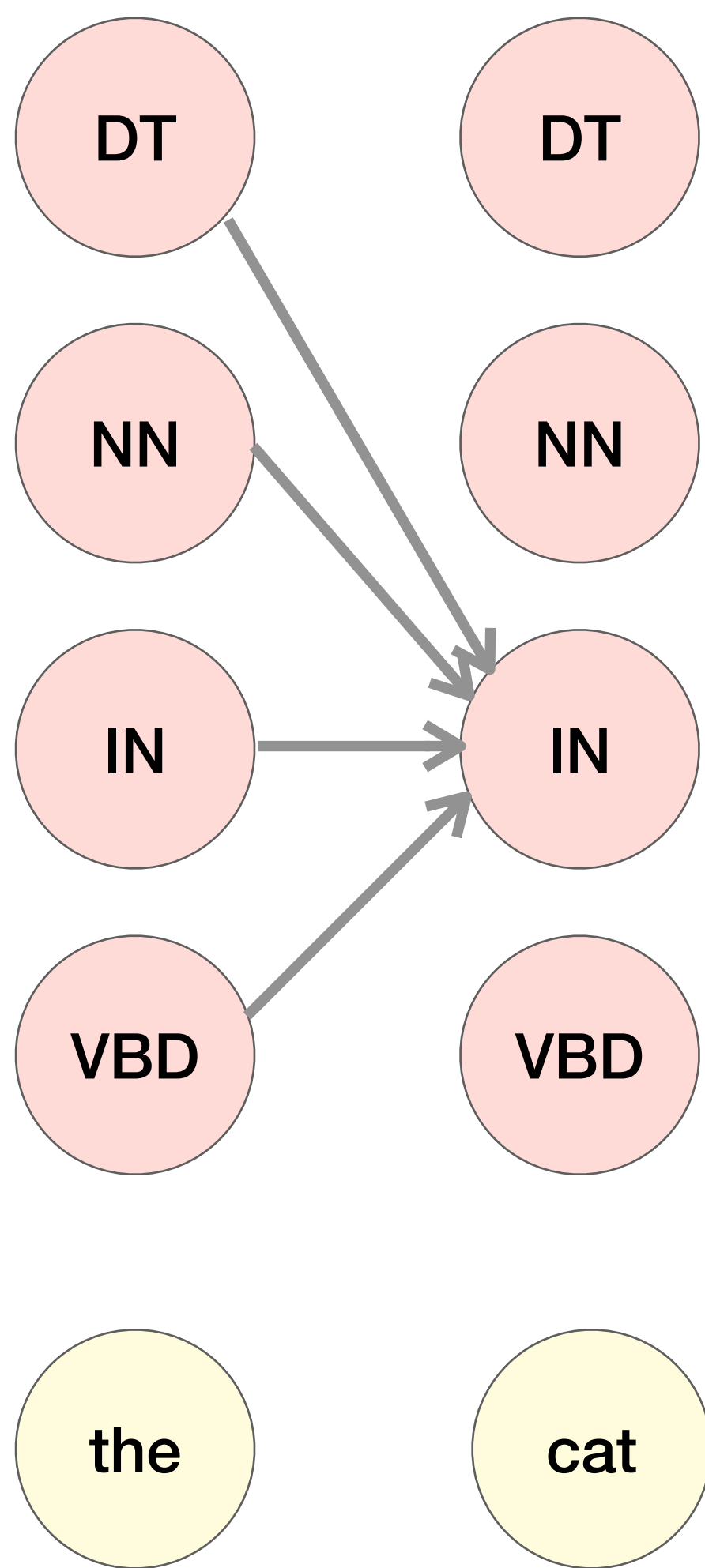
	the	cat	sat	on	the	mat
DT	0.4	0				
NN	0.001	0.064				
IN	0					
VBD	0					

$M[1,K]$

Forward

Viterbi decoding

		s_{t+1}				
		DT	NN	IN	VBD	
s_t	DT	0.05	0.8	0.05	0.1	
	NN	0.05	0.2	0.15	0.6	
	IN	0.5	0.2	0.05	0.25	
	VBD	0.3	0.3	0.3	0.1	
		o_t				
		the	cat	sat	on	mat
s_t	DT	0.5	0	0	0	0
	NN	0.01	0.2	0.01	0.01	0.2
	IN	0	0	0	0.4	0
	VBD	0	0.01	0.1	0.01	0.01



$$M[2,IN] = \max_k M[1,k] P(IN|k) P(\text{cat} | IN) = 0$$

	the	cat	sat	on	the	mat
DT	0.4	0				
NN	0.001	0.064				
IN	0	0				
VBD	0					
		$M[1,K]$				

Forward

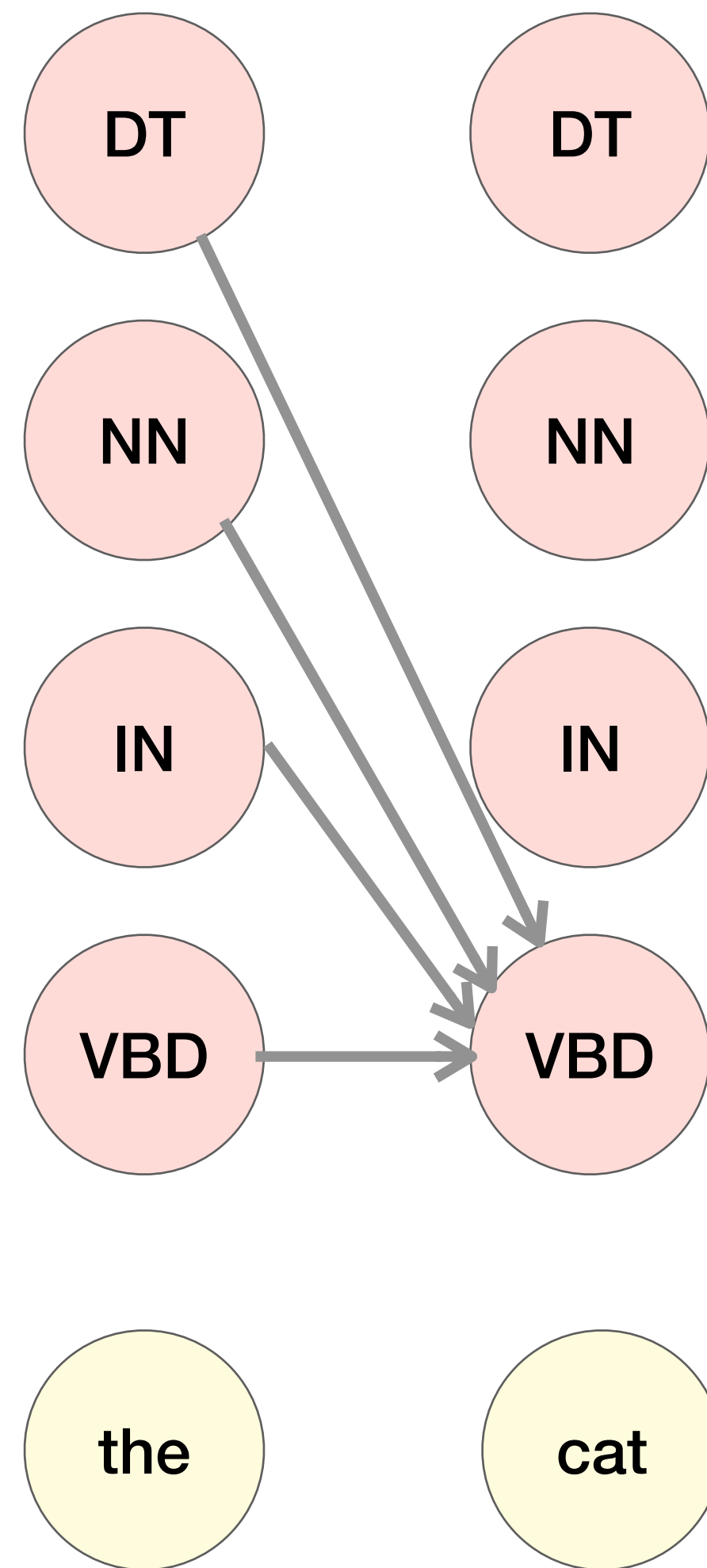
Viterbi decoding

$$S_{t+1}$$

	DT	NN	IN	VBD
DT	0.05	0.8	0.05	0.1
NN	0.05	0.2	0.15	0.6
IN	0.5	0.2	0.05	0.25
VBD	0.3	0.3	0.3	0.1

$$O_t$$

	the	cat	sat	on	mat
DT	0.5	0	0	0	0
NN	0.01	0.2	0.01	0.01	0.2
IN	0	0	0	0.4	0
VBD	0	0.01	0.1	0.01	0.01



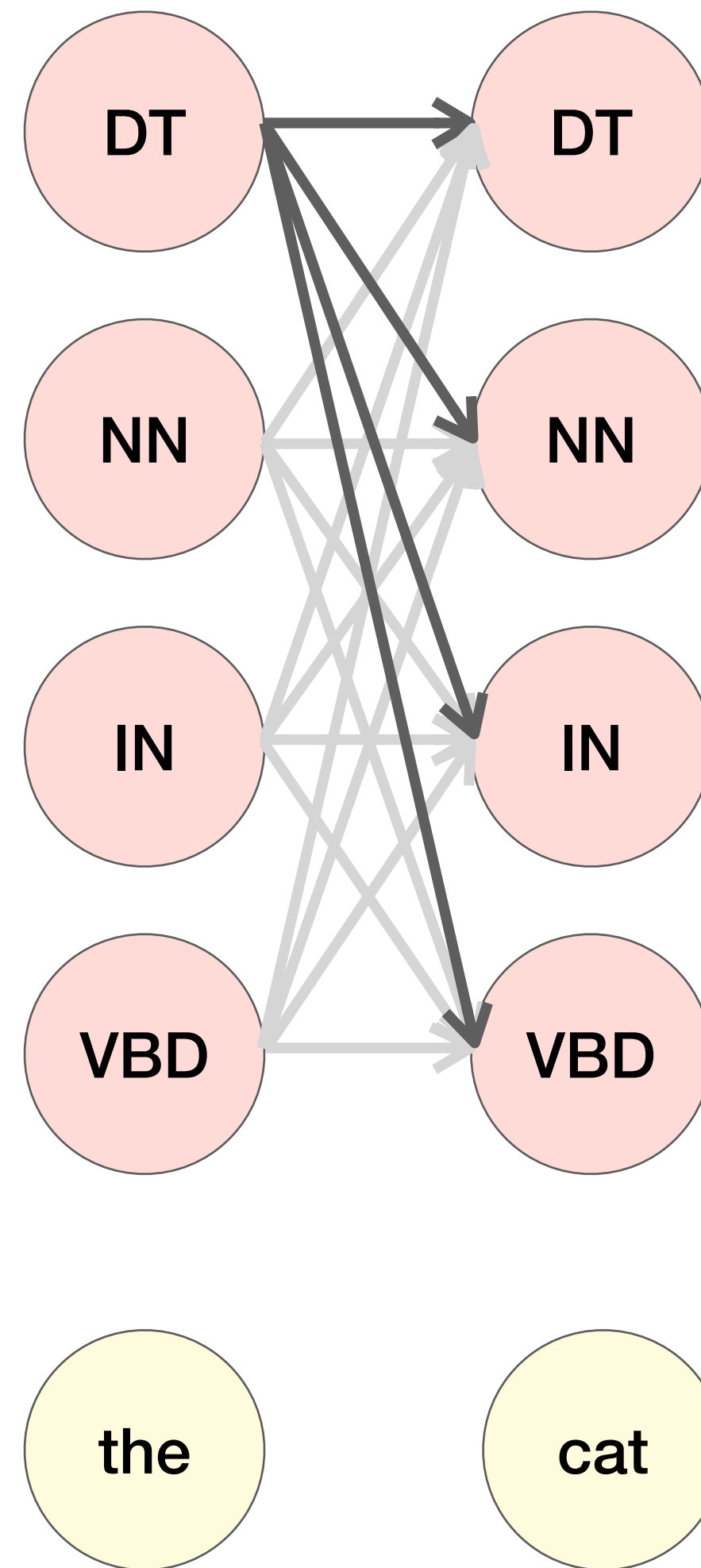
	the	cat	sat	on	the	mat
DT	0.4	0				
NN	0.001	0.064				
IN	0	0				
VBD	0	0.0004				

$M[1,K]$

$$M[2,VBD] = \max_k M[1,k] P(VBD | k) P(\text{cat} | VBD) = 0.0004$$

Forward

Viterbi decoding



$$M[2,DT] = \max_k M[1,k] P(DT|k) P(\text{cat} | DT) = 0$$

$$M[2,NN] = \max_k M[1,k] P(NN|k) P(\text{cat} | NN) = 0.064$$

$$M[2,IN] = \max_k M[1,k] P(IN|k) P(\text{cat} | IN) = 0$$

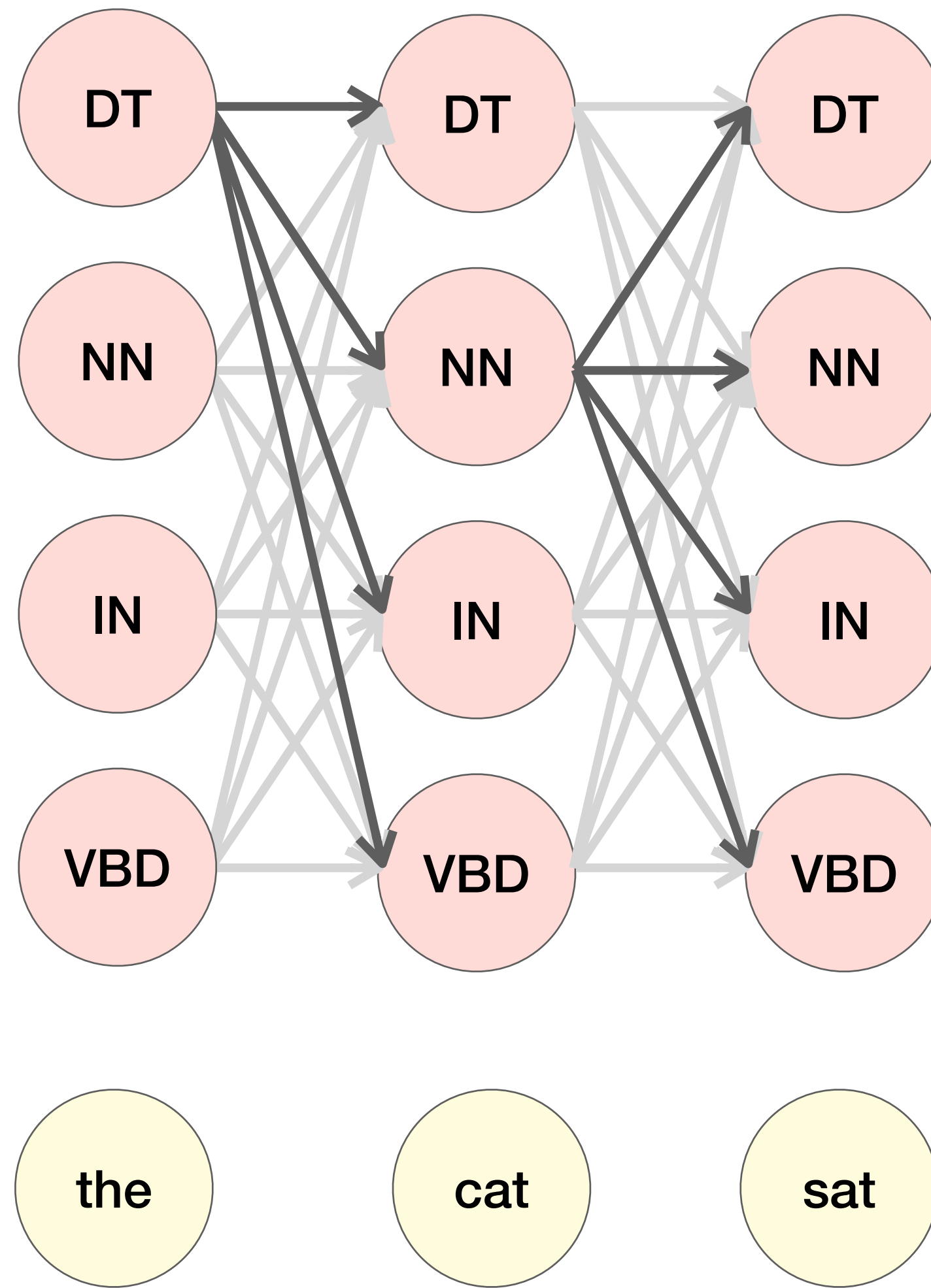
$$M[2,VBD] = \max_k M[1,k] P(VBD|k) P(\text{cat} | VBD) = 0.004$$

Forward

$M[T, K]$

	the	cat	sat	on	the	mat
DT	0.4	0				
NN	0.001	0.064				
IN	0	0				
VBD	0	0.004				

Viterbi decoding



$$M[3,DT] = \max_k M[2,k] P(DT|k) P(\text{sat} | DT) = 0$$

$$M[3,NN] = \max_k M[2,k] P(NN|k) P(\text{sat} | NN) = 0.000128$$

$$M[3,IN] = \max_k M[2,k] P(IN|k) P(\text{sat} | IN) = 0$$

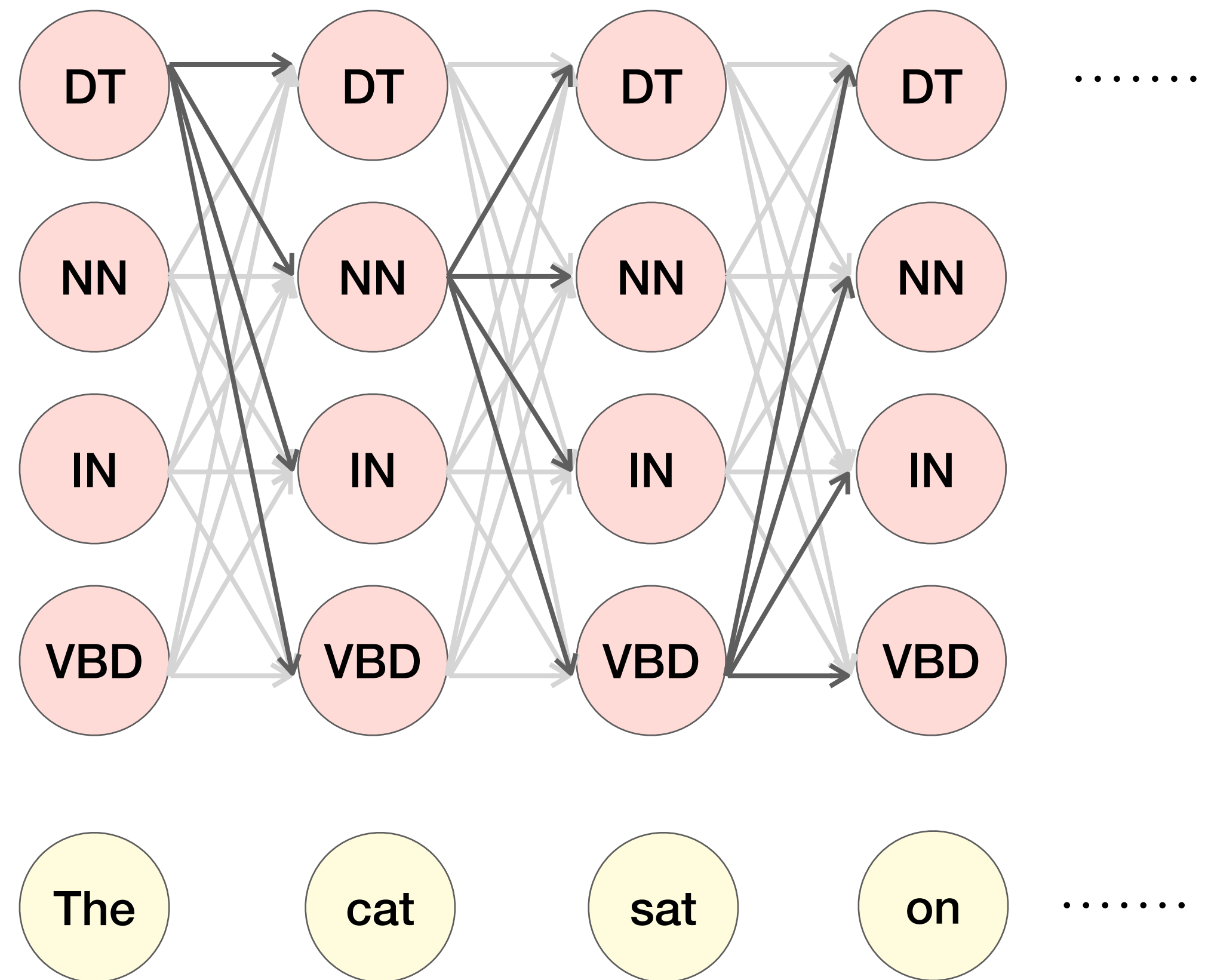
$$M[3,VBD] = \max_k M[2,k] P(VBD|k) P(\text{sat} | VBD) = 0.00384$$

Forward

$$M[T, K]$$

	the	cat	sat	on	the	mat
DT	0.4	0	0			
NN	0.001	0.064	0.000128			
IN	0	0	0			
VBD	0	0.008	0.00384			

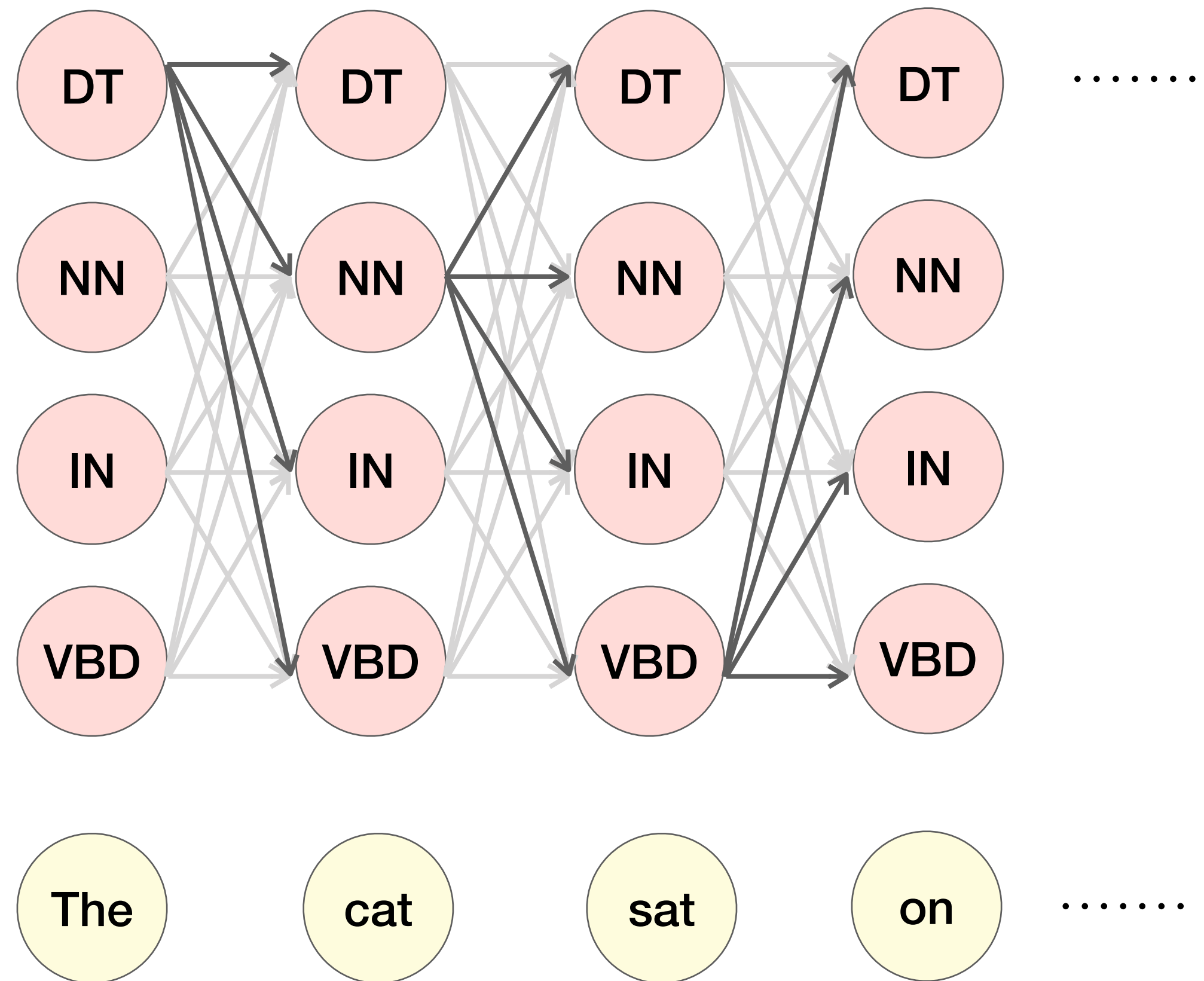
Viterbi decoding



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

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

Viterbi decoding



Time complexity?

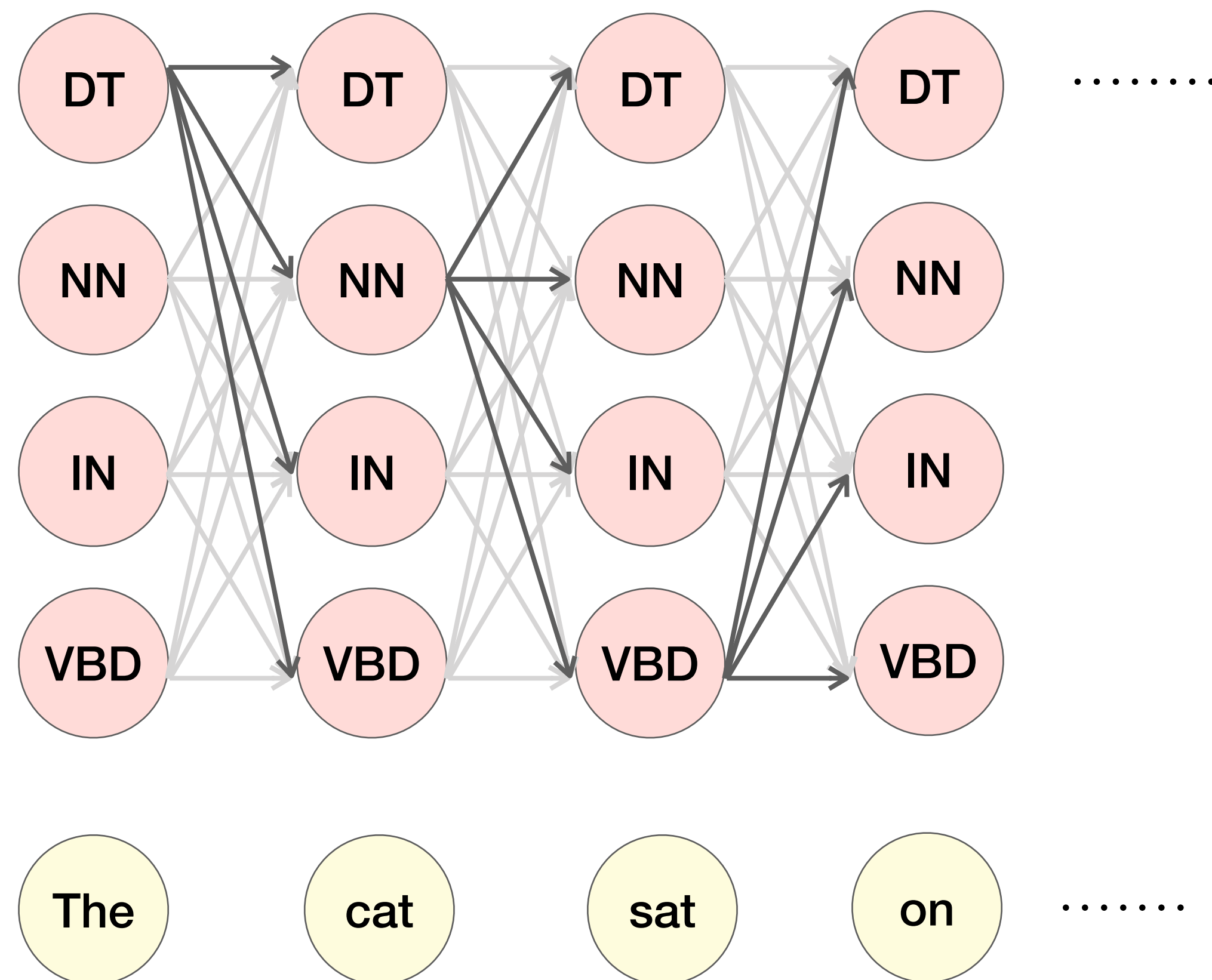
$O(K \times K \times n)$

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

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

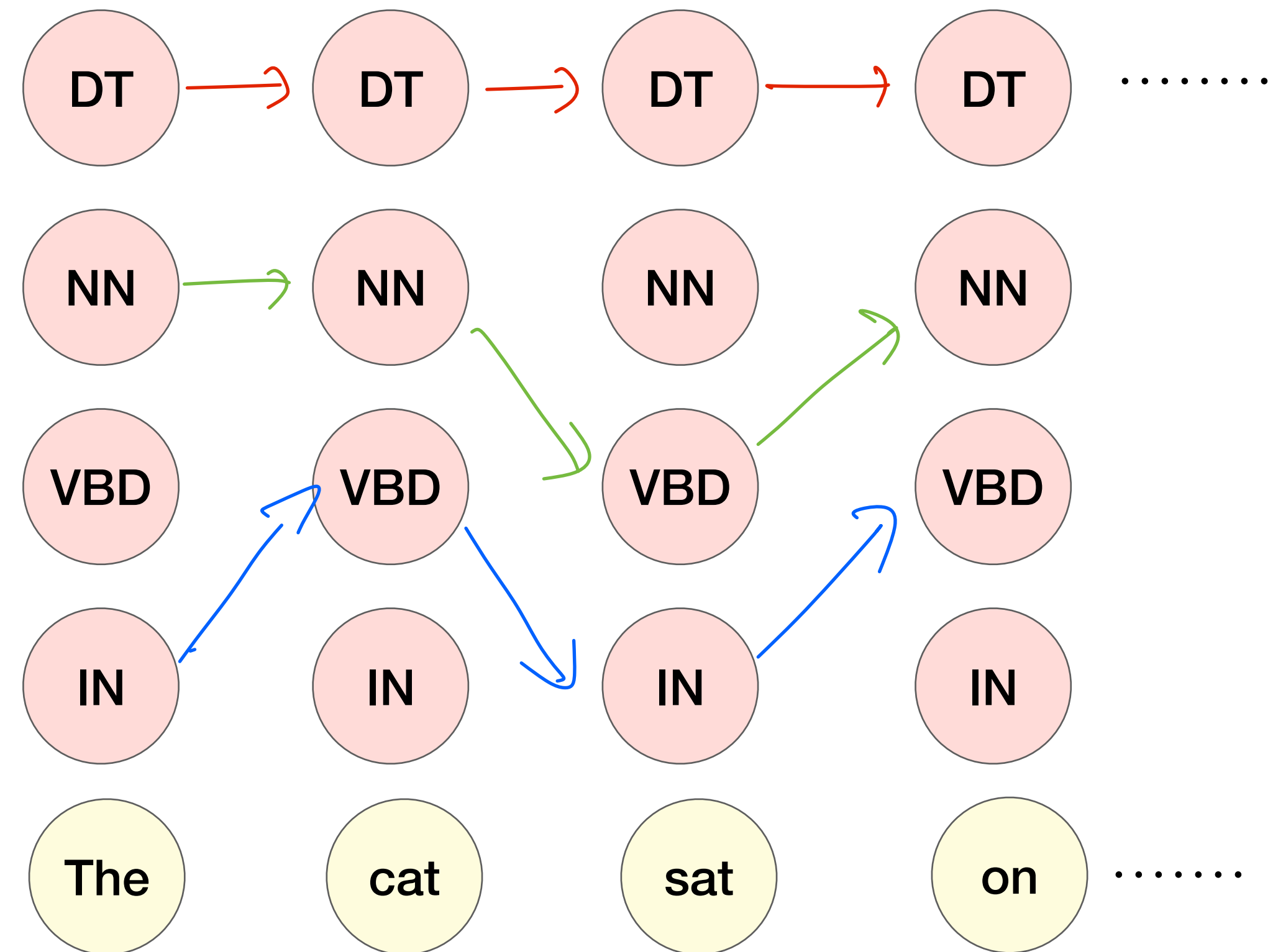
Beam Search

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



Beam Search

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



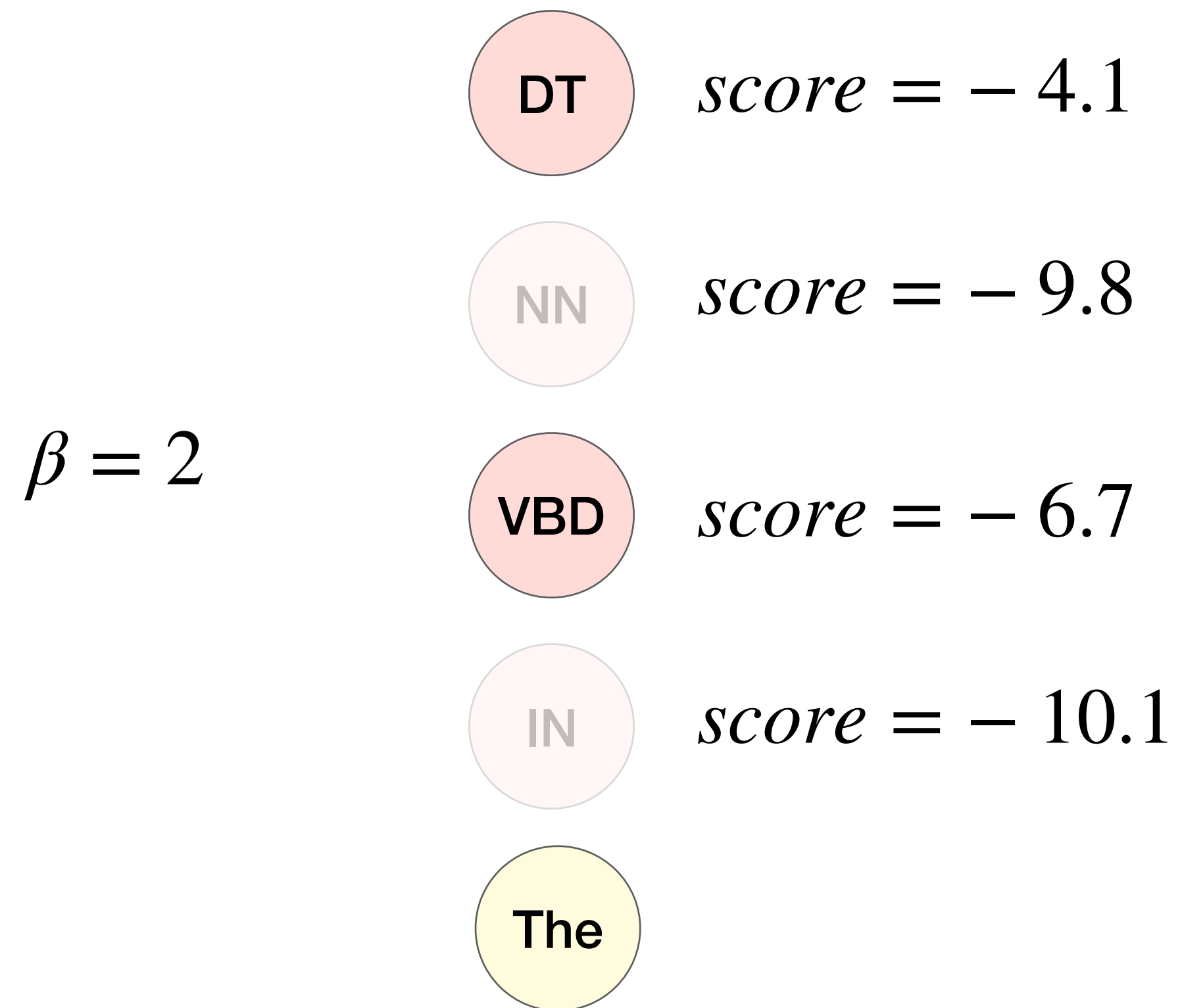
Many paths have very low likelihood!

Beam Search

- If K (number of states) is too large, Viterbi is too expensive!
- Keep a **fixed number** of hypotheses at each point
- Beam width, β

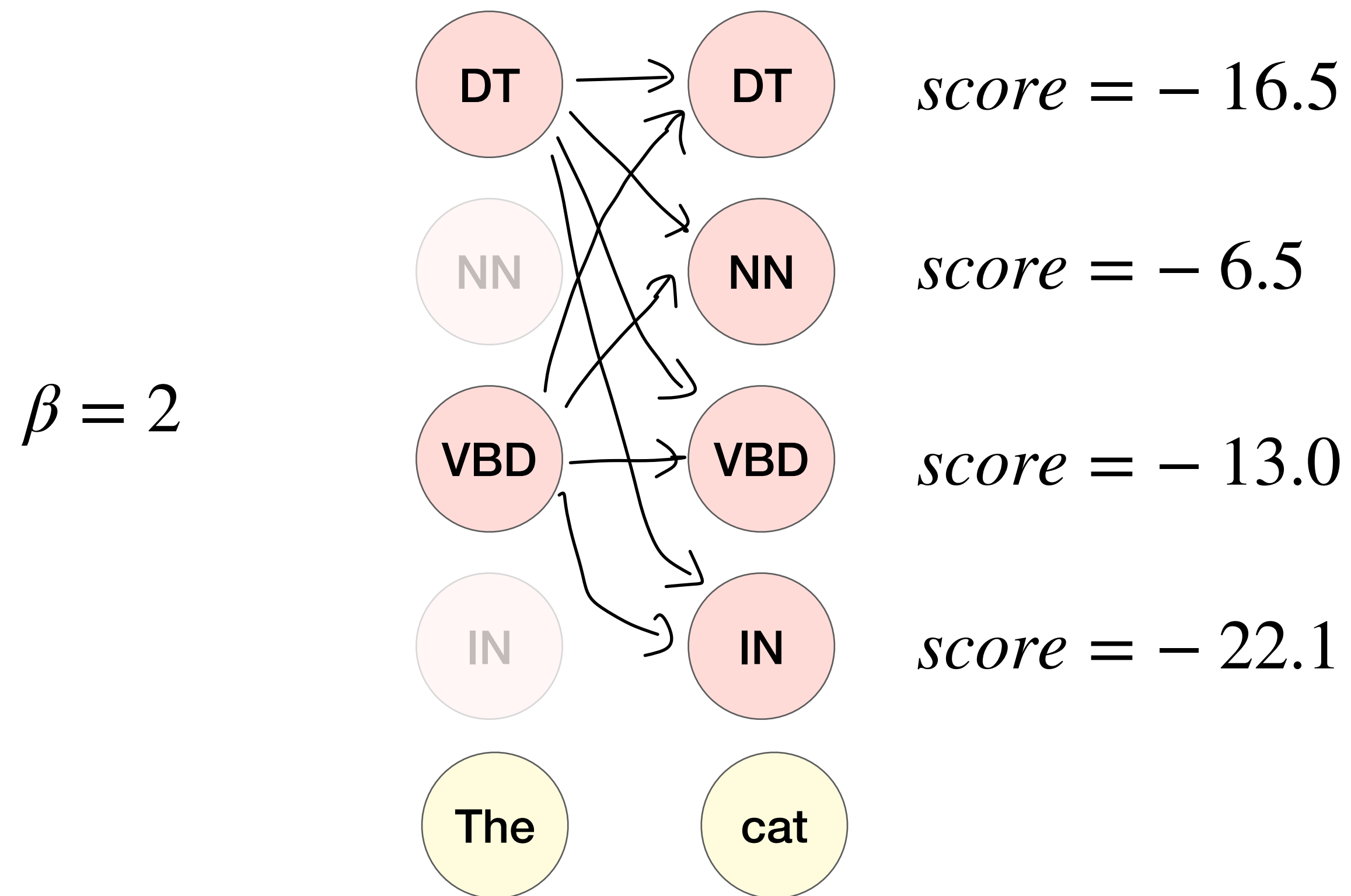
Beam Search

- Keep a fixed number of hypotheses at each point



Beam Search

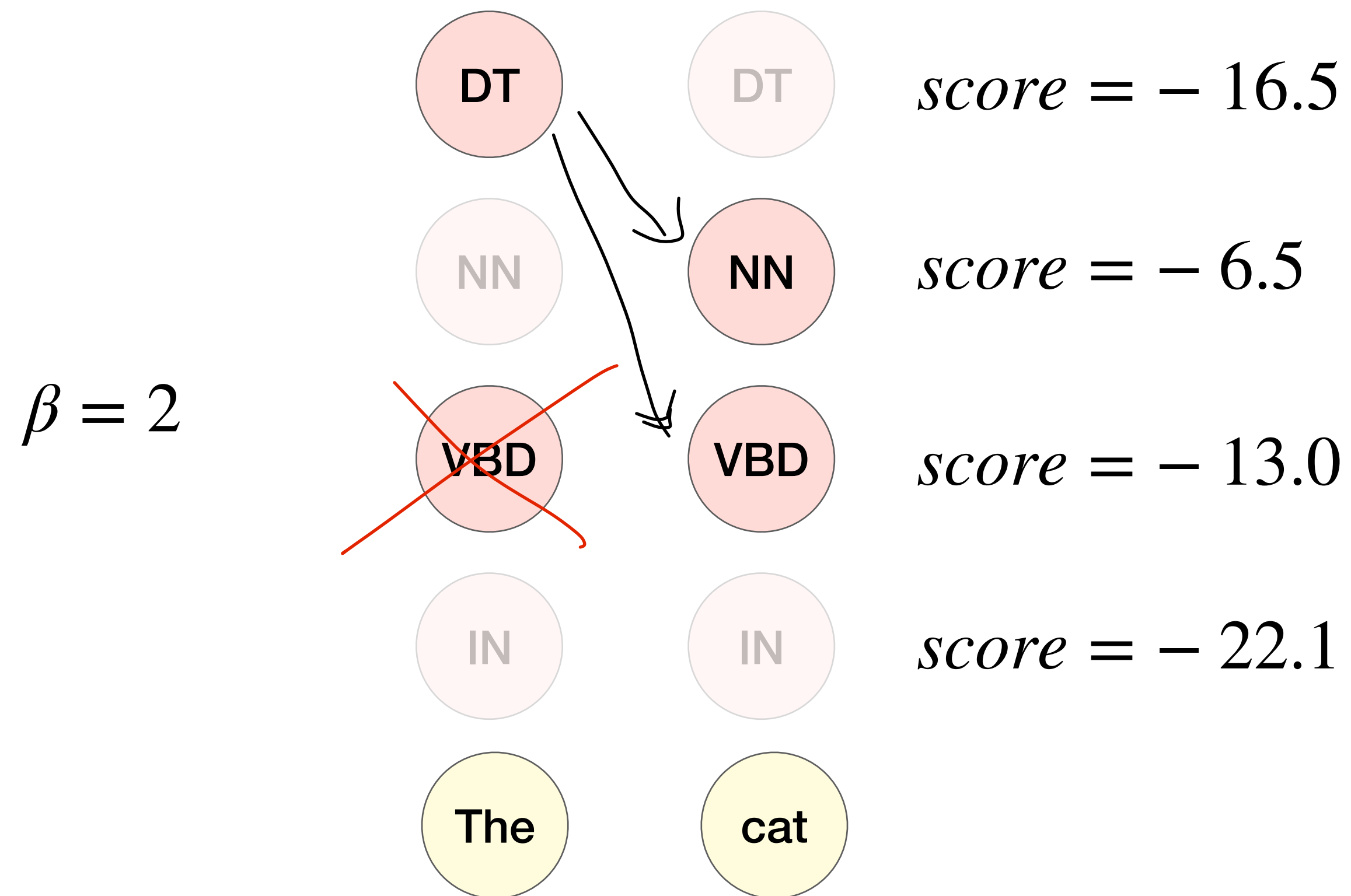
- Keep a fixed number of hypotheses at each point



Step 1: Expand all partial sequences in current beam

Beam Search

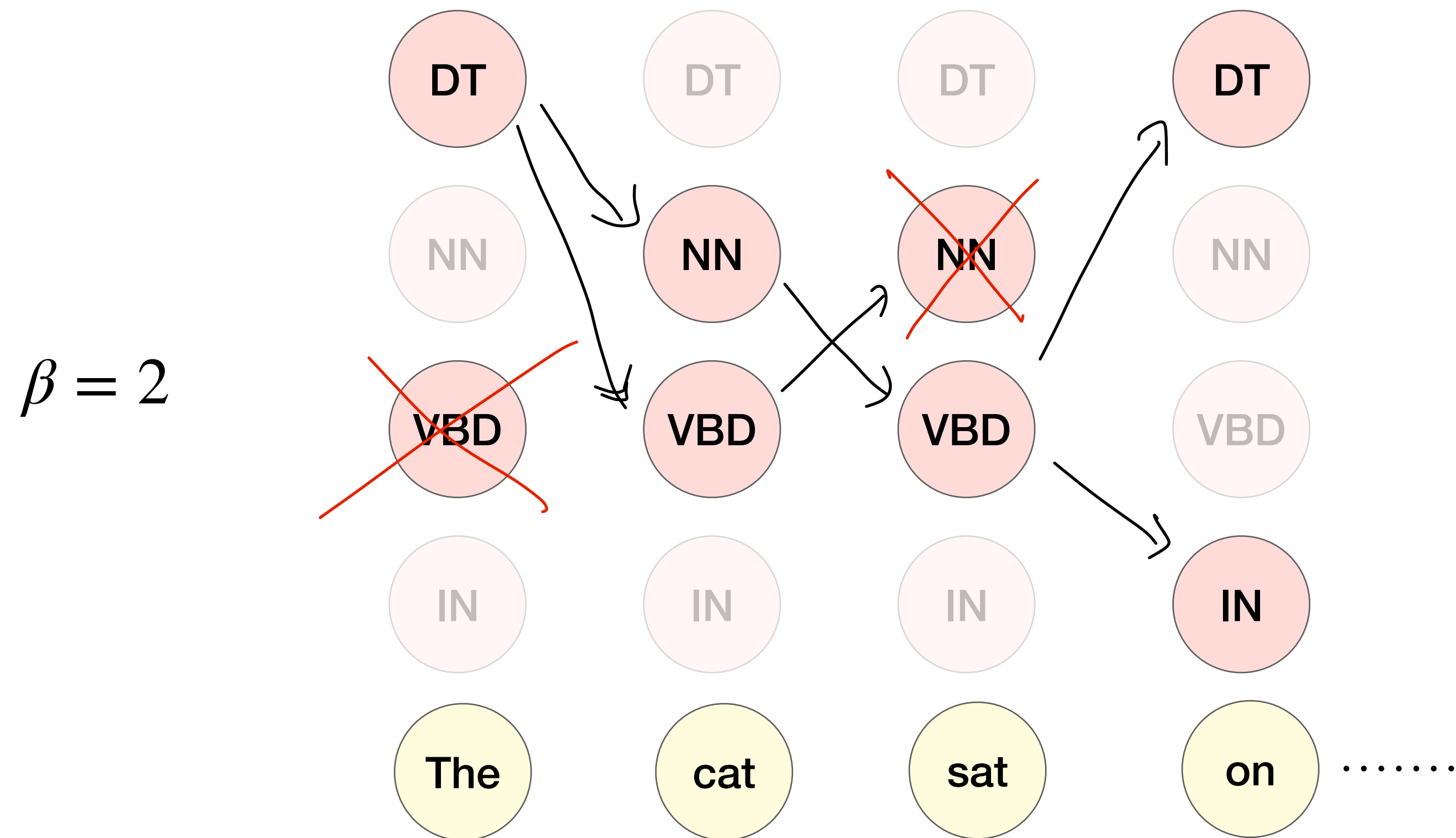
- Keep a fixed number of hypotheses at each point



Step 2: Prune set back to top β sequences

Beam Search

- Keep a fixed number of hypotheses at each point



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

Beam Search

- If K (number of states) is too large, Viterbi is too expensive!
- Keep a fixed number of hypotheses at each point
 - Beam width, β
- Trade-off computation for (some) accuracy

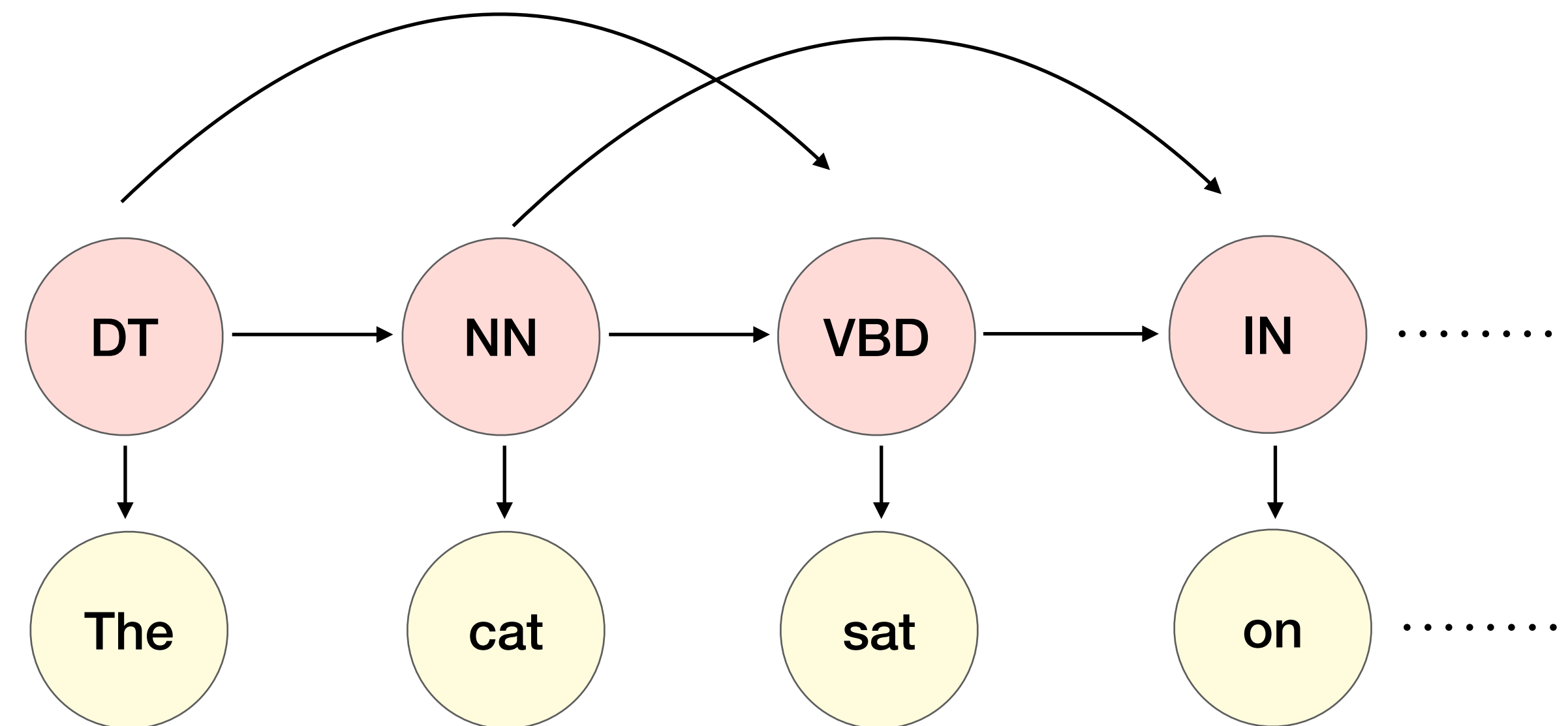
Time complexity?

$$\beta \times K \times T$$

Beyond unigram HMMs

Beyond bigrams

- Real-world HMM taggers have more relaxed assumptions
- Trigram HMM: $P(s_{t+1} | s_1, s_2, \dots, s_t) \approx P(s_{t+1} | s_{t-1}, s_t)$



Pros?

Cons?

HMMs for language modeling

- Language modeling: estimate probability of sentence

$P(\text{the, cat, sat, on, the, mat}) = ??$

- Need to sum over the probabilities of the possible states

$$P(O) = P(o_1, o_2, \dots, o_n) = \sum_{s_1, \dots, s_n} \prod_{t=1}^n P(o_t | s_t) P(s_t | s_{t-1})$$

- Use Viterbi-like algorithm, but take sum instead of max!
 - Known as the “Forward” algorithm

Generative vs Discriminative

- HMM is a *generative* model
- Can we model $P(s_1, \dots, s_n \mid o_1, \dots, o_n)$ directly?

Generative

Naive Bayes:

$$P(c)P(d \mid c)$$

HMM:

$$P(s_1, \dots, s_n)P(o_1, \dots, o_n \mid s_1, \dots, s_n)$$

Discriminative

Logistic Regression:

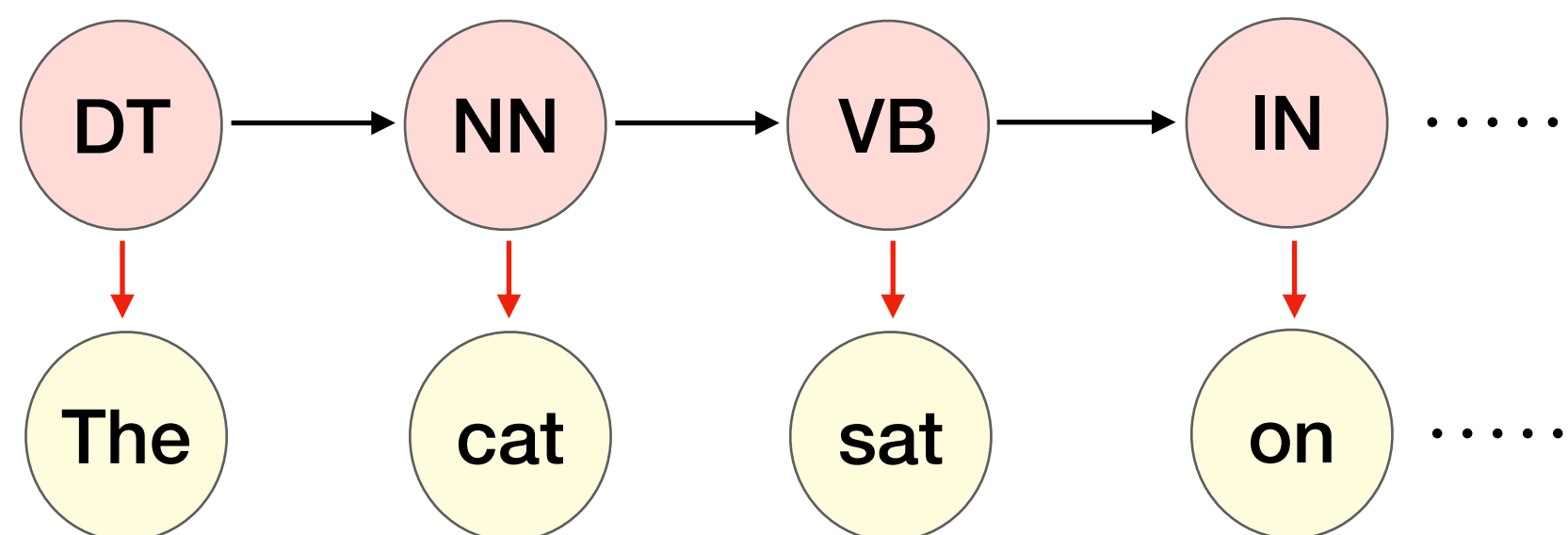
$$P(c \mid d)$$

MEMM:

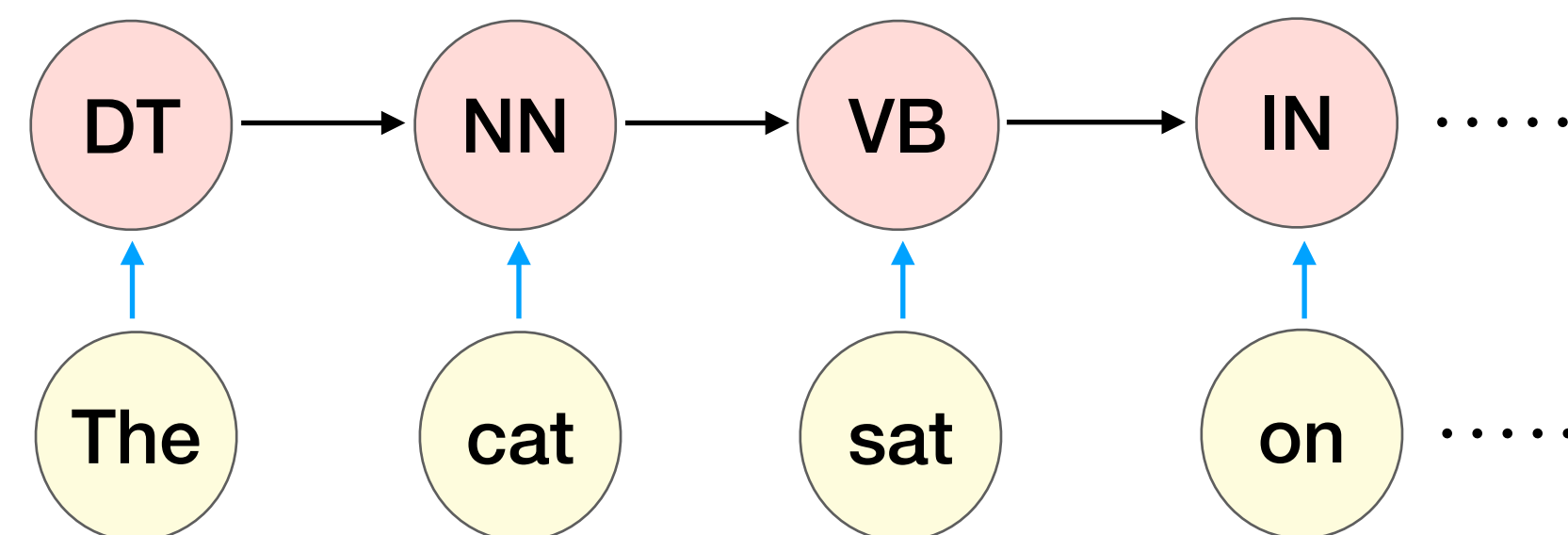
$$P(s_1, \dots, s_n \mid o_1, \dots, o_n)$$

No factorization

MEMM



HMM



MEMM

- Compute the posterior directly:
No factorization
Into emission and transition probabilities

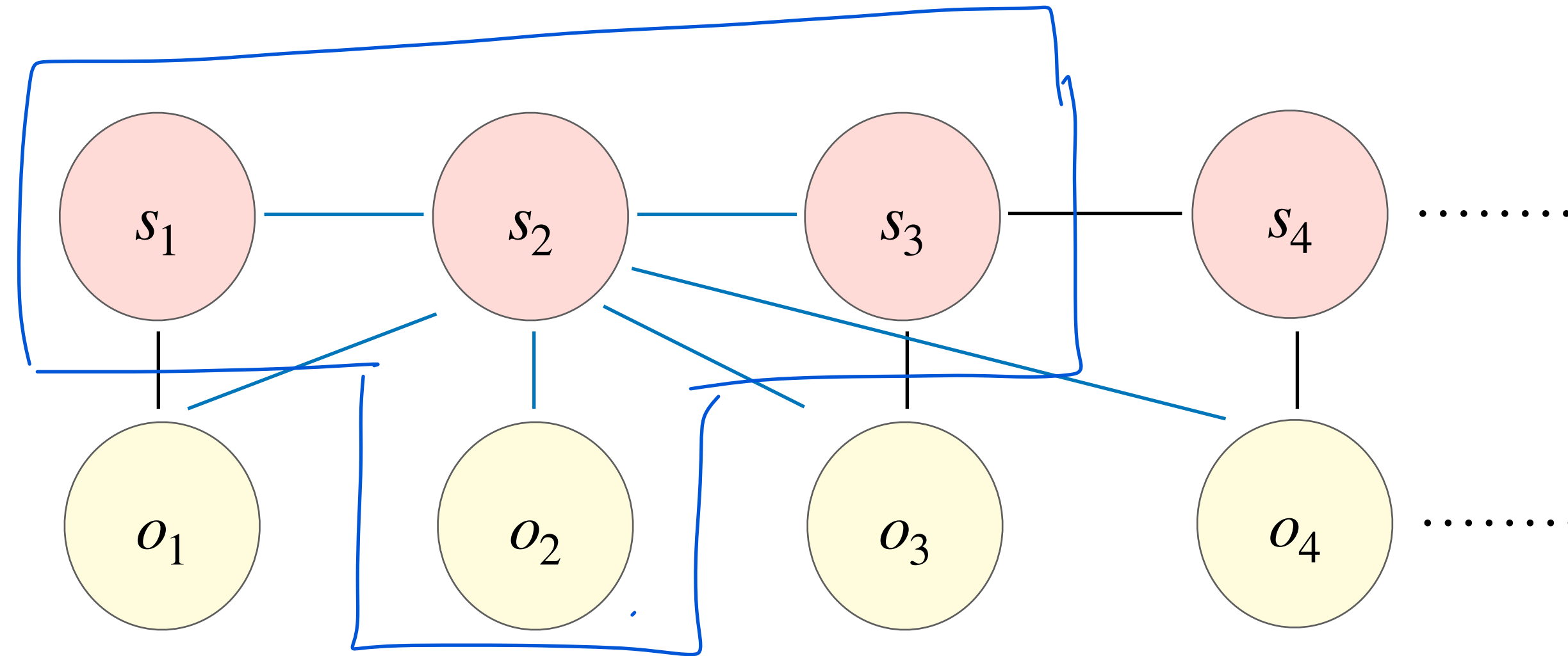
$$\hat{S} = \arg \max_S P(S | O) = \arg \max_S \prod_i P(s_i | o_i, s_{i-1})$$

- Use features: $P(s_i | o_i, s_{i-1}) \propto \exp(w \cdot f(s_i, o_i, s_{i-1}))$

Features

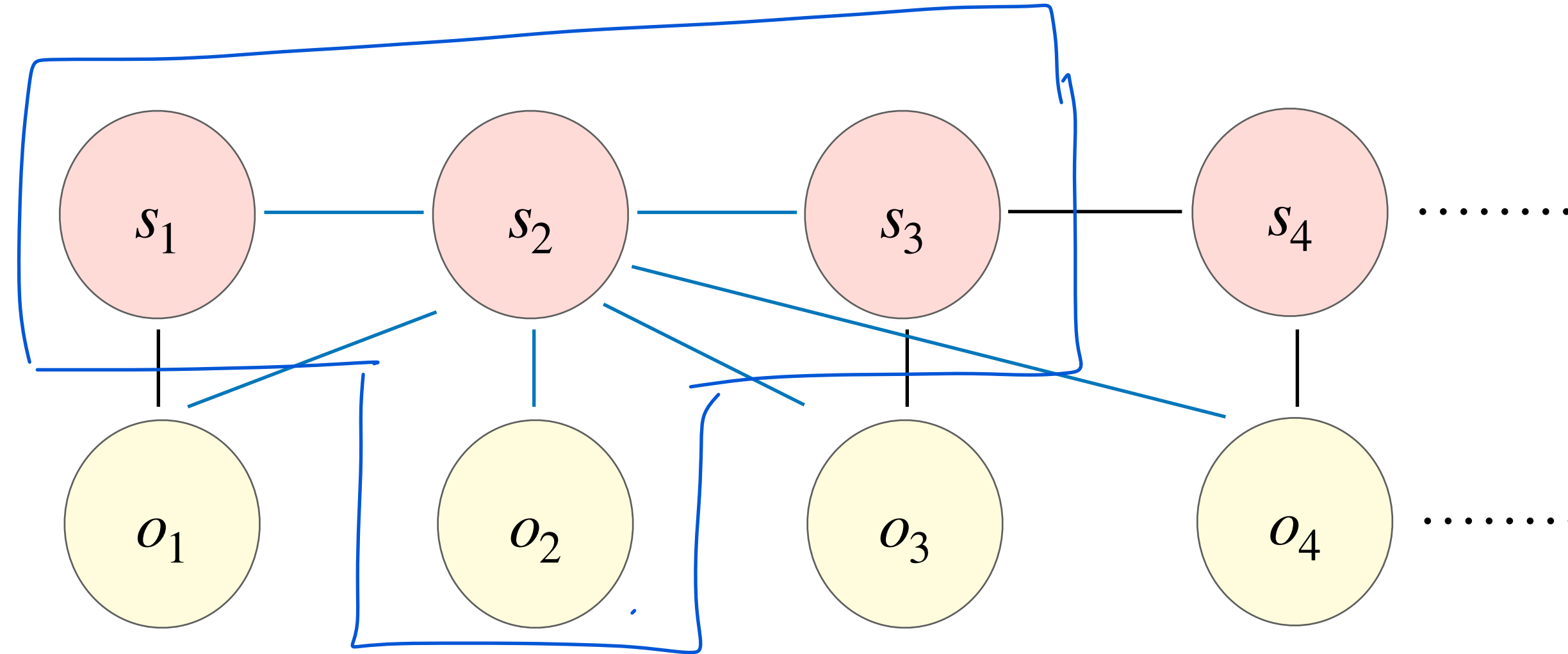
weights

Conditional Random Field (advanced)



- Compute **log-linear** functions over cliques
- Less independence assumptions
- Ex: $P(s_t \mid \text{everything else}) \propto \exp(w \cdot f(s_{t-1}, s_t, s_{t+1}, O))$

Conditional Random Field (advanced)

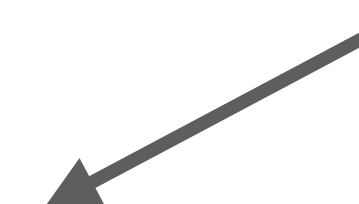


- Define feature functions f
- Compute score $w \cdot f(s_{t-1}, s_t, s_{t+1}, O)$

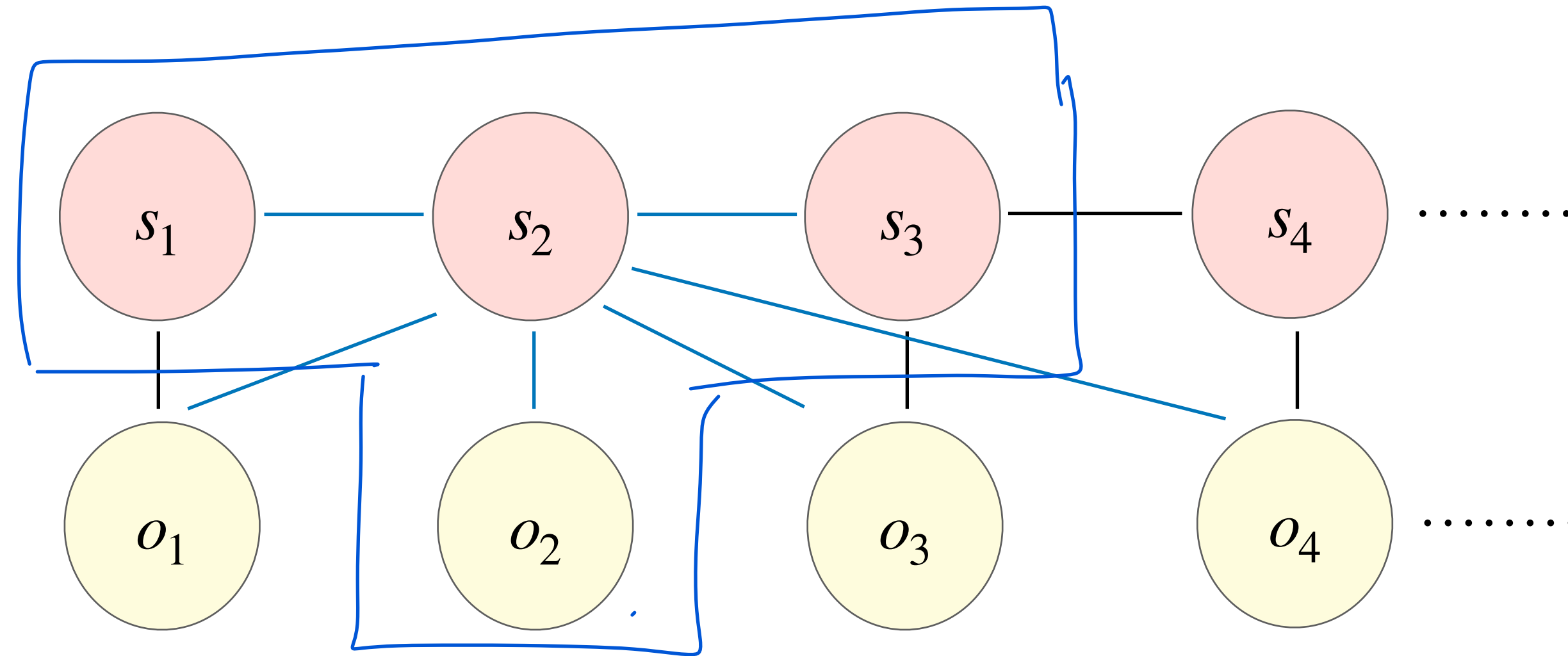
- Normalize
$$P(s_t | O) = \frac{\exp(w \cdot f(s_{t-1}, s_t, s_{t+1}, O))}{\sum_s \exp(w \cdot f(s_{t-1}, s_t, s_{t+1}, O))}$$

Global vs Local
normalization

Global normalization
can be expensive



Conditional Random Field (advanced)



- Decoding: Best sequence can be found using the Viterbi algorithm
- Training: Find parameters using variant of Forward algorithm (or SGD)
- Can be used on top of other model outputs to refine output for consistency

Summary

- Sequence modeling - POS tagging, Named Entity Recognition
- Hidden Markov Models (HMM)
 - Generative model
 - Markov assumption: assume current state only depend on last few states
 - Labels are considered “hidden states”
 - Training
 - If training data includes labels, MLE is simple
 - More generally (partially observed/no labels), use Expectation-Maximation (EM)
 - Decoding
 - Greedy, Viterbi (exact), Beam search
- CRFs - Discriminative, log-linear models for sequence modelling

Maximum Entropy Markov Models

(extra content - not covered)

Generative vs Discriminative

- HMM is a *generative* model
- Can we model $P(s_1, \dots, s_n \mid o_1, \dots, o_n)$ directly?

Generative

Naive Bayes:

$$P(c)P(d \mid c)$$

HMM:

$$P(s_1, \dots, s_n)P(o_1, \dots, o_n \mid s_1, \dots, s_n)$$

Discriminative

Logistic Regression:

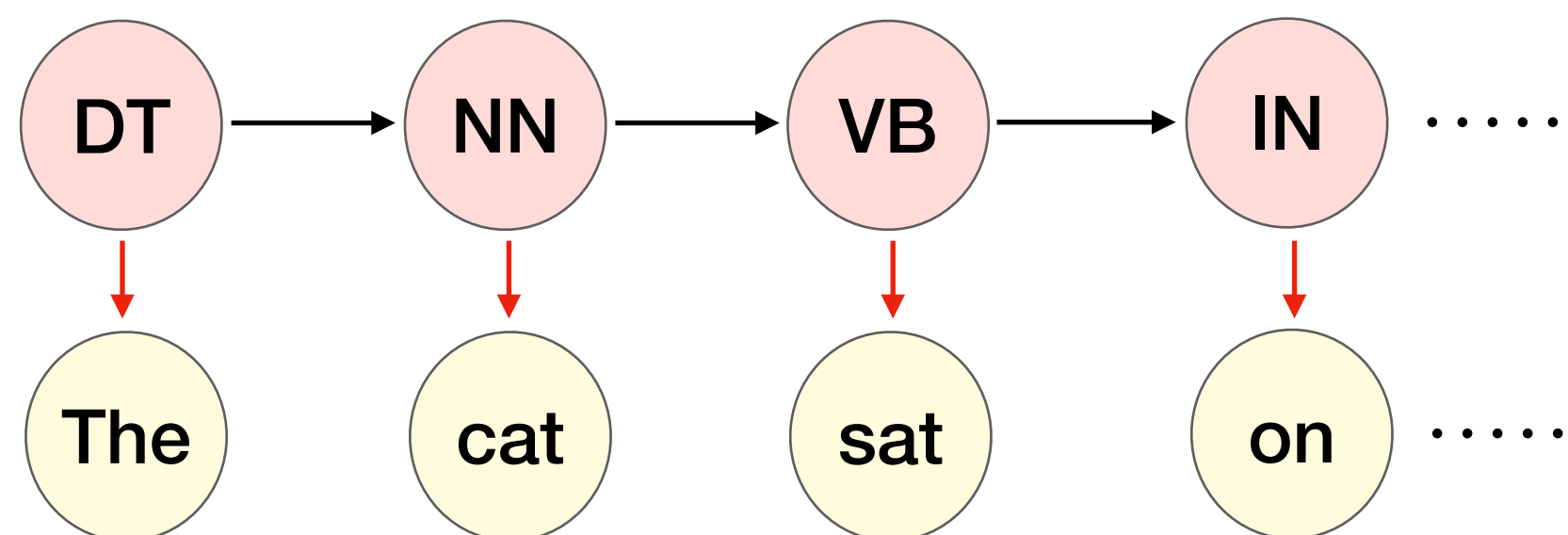
$$P(c \mid d)$$

MEMM:

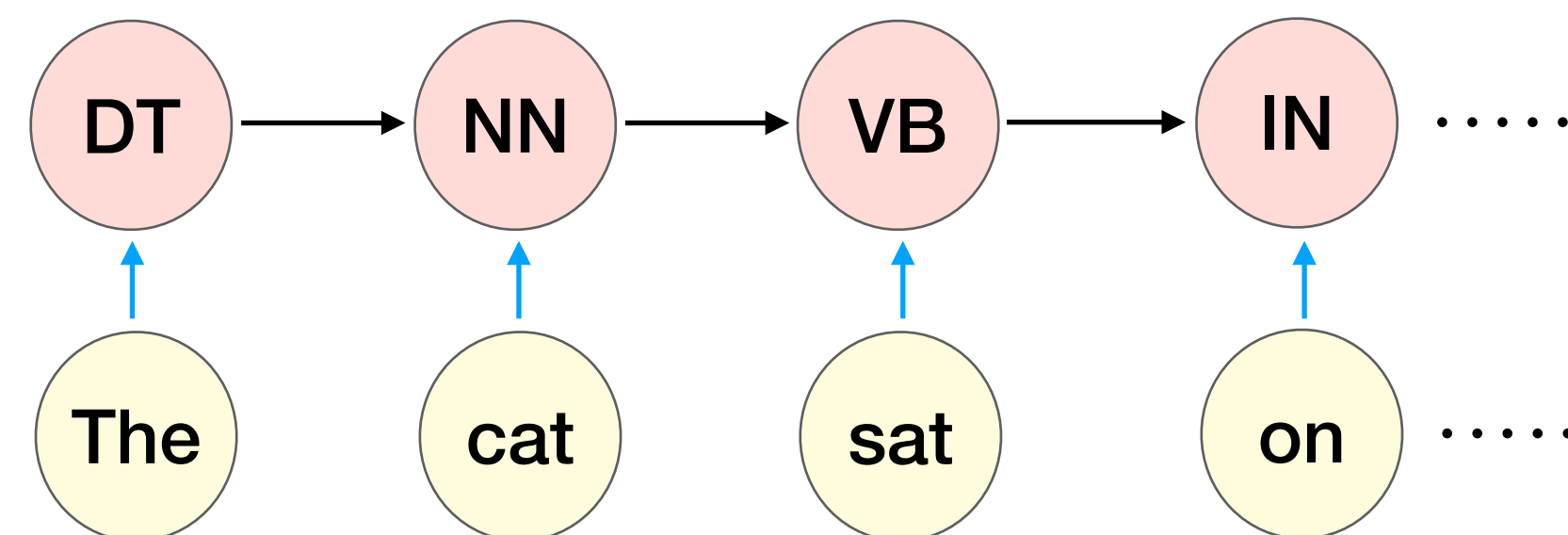
$$P(s_1, \dots, s_n \mid o_1, \dots, o_n)$$

No factorization

MEMM



HMM



MEMM

No factorization

- Compute the posterior directly:

Into emission and transition probabilities

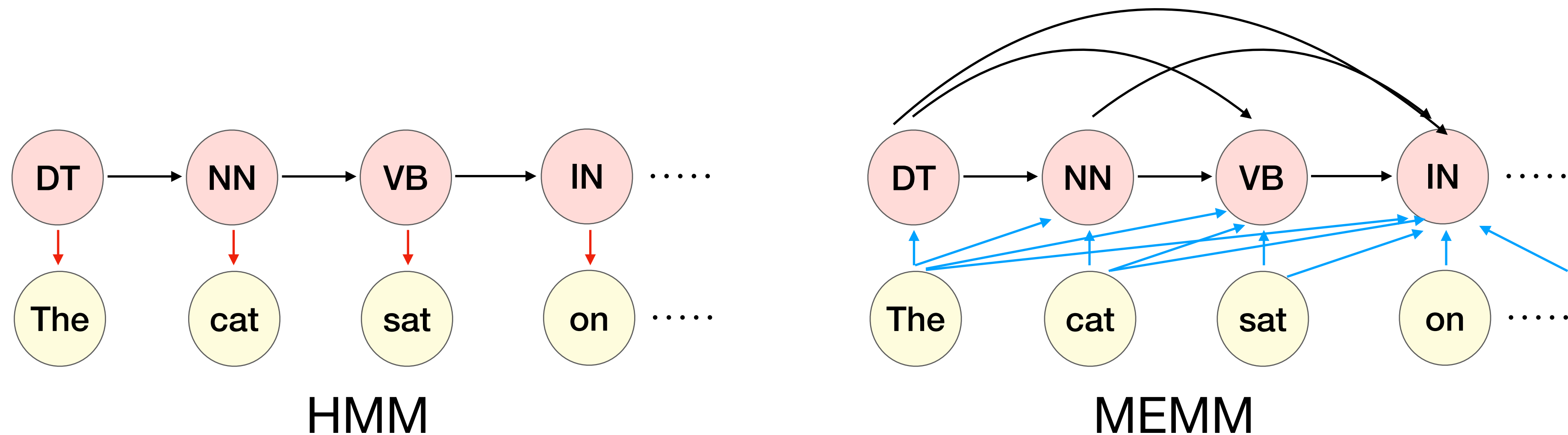
$$\hat{S} = \arg \max_S P(S | O) = \arg \max_S \prod_i P(s_i | o_i, s_{i-1})$$

- Use features: $P(s_i | o_i, s_{i-1}) \propto \exp(w \cdot f(s_i, o_i, s_{i-1}))$

Features

weights

MEMM

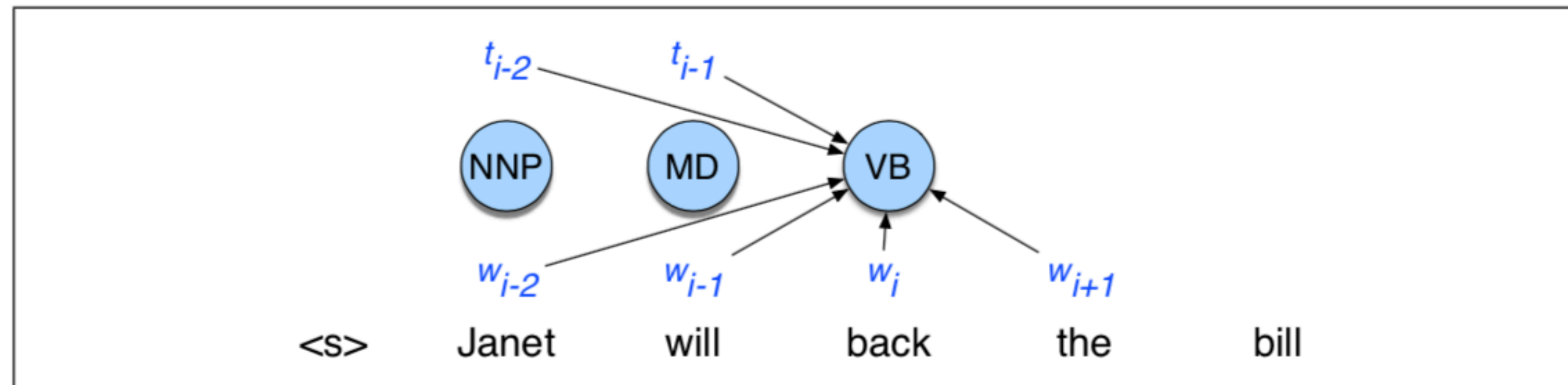


- In general, we can use all observations and all previous states:

$$\hat{S} = \arg \max_S P(S | O) = \arg \max_S \prod_i P(s_i | o_n, o_{i-1}, \dots, o_1, s_{i-1}, \dots, s_1)$$

$$P(s_i | s_{i-1}, \dots, s_1, O) \propto \exp(w \cdot f(s_i, s_{i-1}, \dots, s_1, O))$$

Features in an MEMM



t = tags
w = words
(observations)

Figure 8.13 An MEMM for part-of-speech tagging showing the ability to condition on more features.

$\langle t_i, w_{i-2} \rangle, \langle t_i, w_{i-1} \rangle, \langle t_i, w_i \rangle, \langle t_i, w_{i+1} \rangle, \langle t_i, w_{i+2} \rangle$
 $\langle t_i, t_{i-1} \rangle, \langle t_i, t_{i-2}, t_{i-1} \rangle,$
 $\langle t_i, t_{i-1}, w_i \rangle, \langle t_i, w_{i-1}, w_i \rangle, \langle t_i, w_i, w_{i+1} \rangle,$

Feature templates

$t_i = \text{VB}$ and $w_{i-2} = \text{Janet}$
 $t_i = \text{VB}$ and $w_{i-1} = \text{will}$
 $t_i = \text{VB}$ and $w_i = \text{back}$
 $t_i = \text{VB}$ and $w_{i+1} = \text{the}$
 $t_i = \text{VB}$ and $w_{i+2} = \text{bill}$
 $t_i = \text{VB}$ and $t_{i-1} = \text{MD}$
 $t_i = \text{VB}$ and $t_{i-1} = \text{MD}$ and $t_{i-2} = \text{NNP}$
 $t_i = \text{VB}$ and $w_i = \text{back}$ and $w_{i+1} = \text{the}$

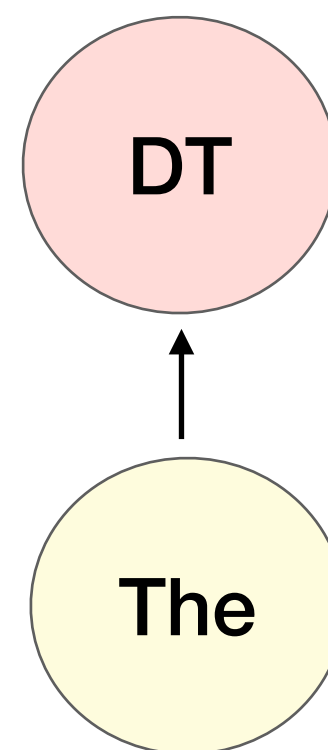
Features

MEMMs: Decoding

$$\hat{S} = \arg \max_S P(S | O) = \arg \max_S \prod_i P(s_i | o_i, s_{i-1})$$

(assume features only on previous time step and current obs)

- Greedy decoding:

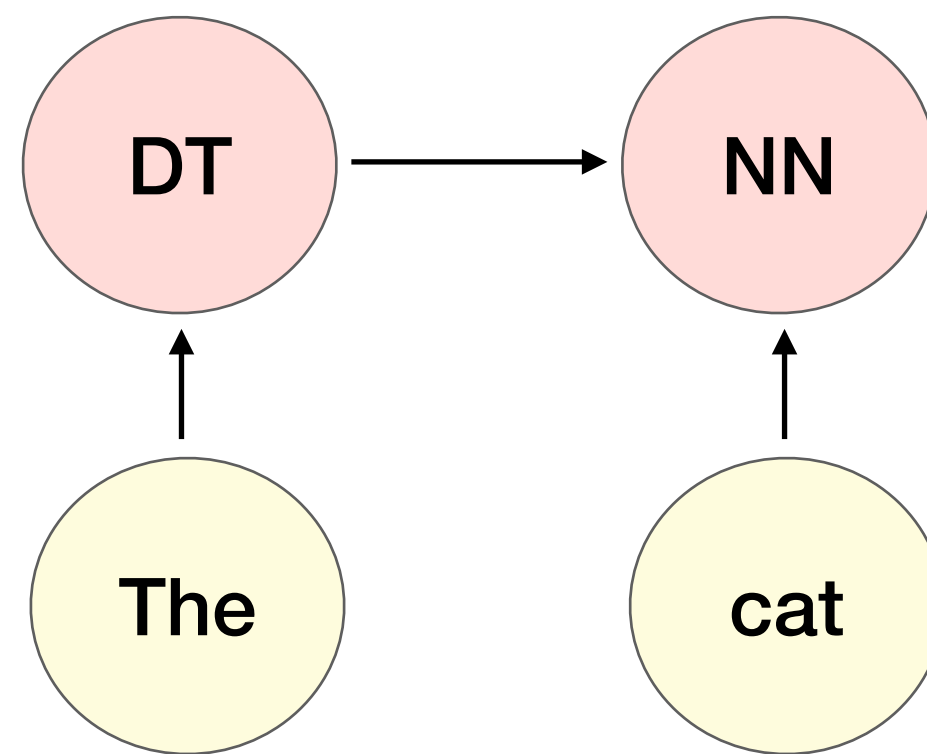


$$\begin{aligned} \hat{s}_1 &= \arg \max_s P(s | \text{The}) \\ &= \text{DT} \end{aligned}$$

MEMMs: Decoding

$$\hat{S} = \arg \max_S P(S | O) = \arg \max_S \prod_i P(s_i | o_i, s_{i-1})$$

- Greedy decoding:

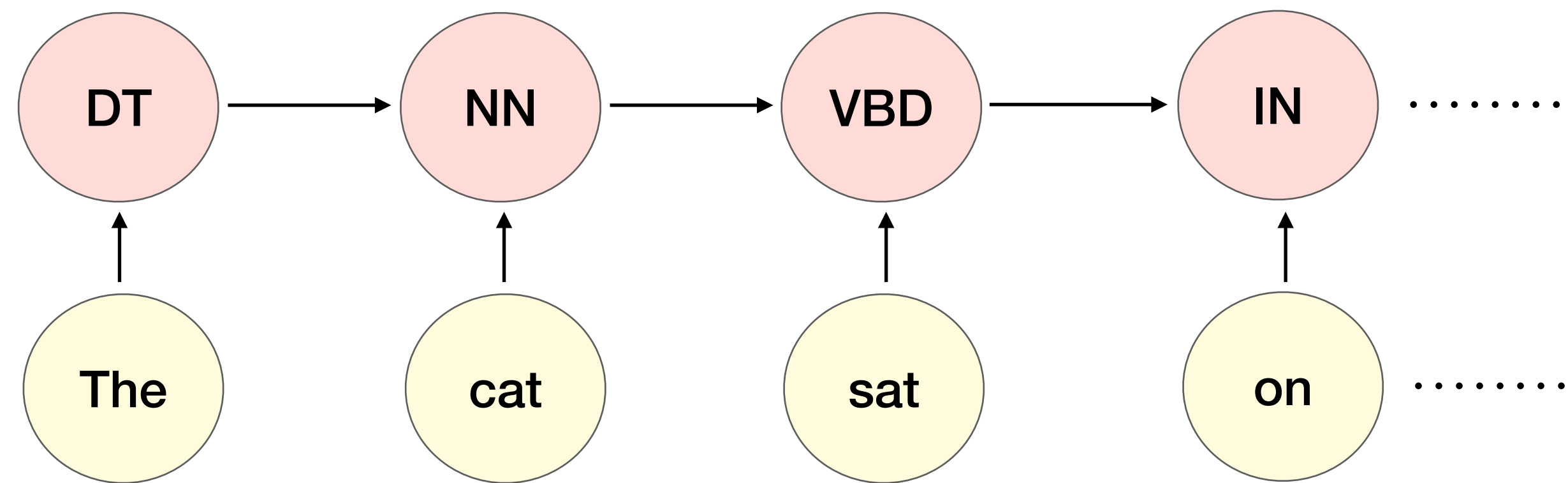


$$\begin{aligned} \hat{s}_2 &= \arg \max_S P(s | \text{cat}, \text{DT}) \\ &= \text{NN} \end{aligned}$$

MEMMs: Decoding

$$\hat{S} = \arg \max_S P(S | O) = \arg \max_S \prod_i P(s_i | o_i, s_{i-1})$$

- Greedy decoding:



$$\forall t, \quad \hat{s}_{t+1} = \arg \max_S P(s | o_{t+1}, \hat{s}_t)$$

MEMMs: Decoding

$$\hat{S} = \arg \max_S P(S | O) = \arg \max_S \prod_i P(s_i | o_i, s_{i-1})$$

- Greedy decoding
- Viterbi decoding:

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

↑
DP Lattice

↑
states

↑
timesteps

MEMM: Learning

- Gradient descent: similar to logistic regression!

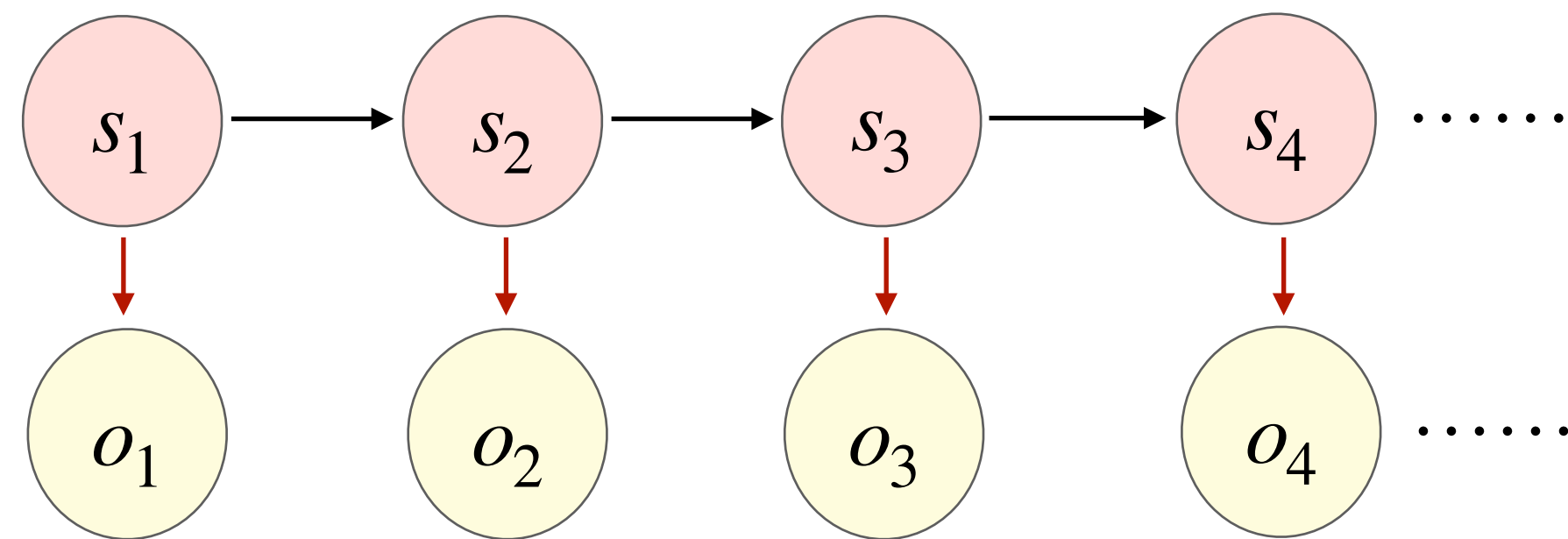
$$P(s_i | s_1, \dots, s_{i-1}, O) = \frac{\exp(w \cdot f(s_1, \dots, s_i, O))}{\sum_{s'} \exp(w \cdot f(s_1, \dots, s', O))}$$

- Given: pairs of (S, O) where each $S = \langle s_1, s_2, \dots, s_n \rangle$

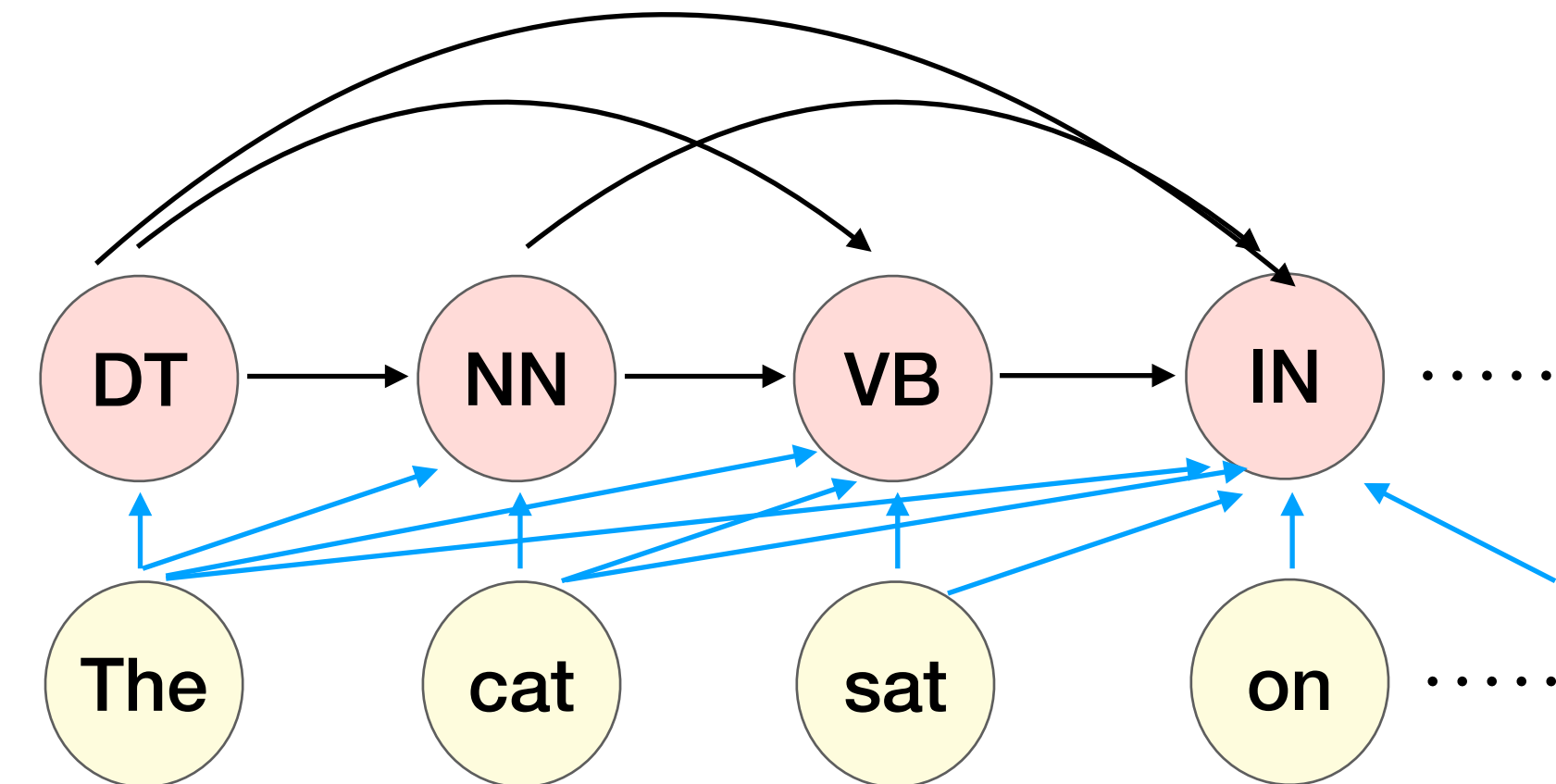
Loss for one sequence, $L = - \sum_i \log P(s_i | s_1, \dots, s_{i-1}, O)$

- Compute gradients with respect to weights w and update

Label bias



HMM



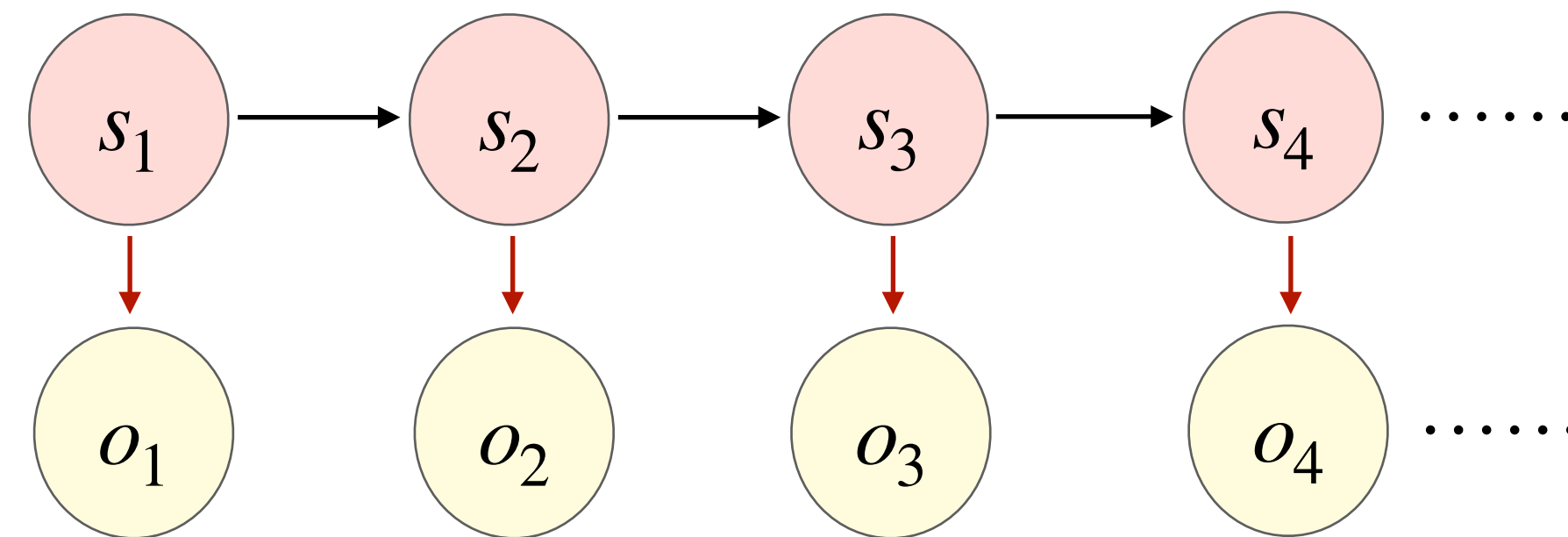
MEMM

Both HMM and MEMM assume left-to-right processing

Why can this be undesirable?

Low entropy transitions between labels may override the effect of observations

Bidirectionality



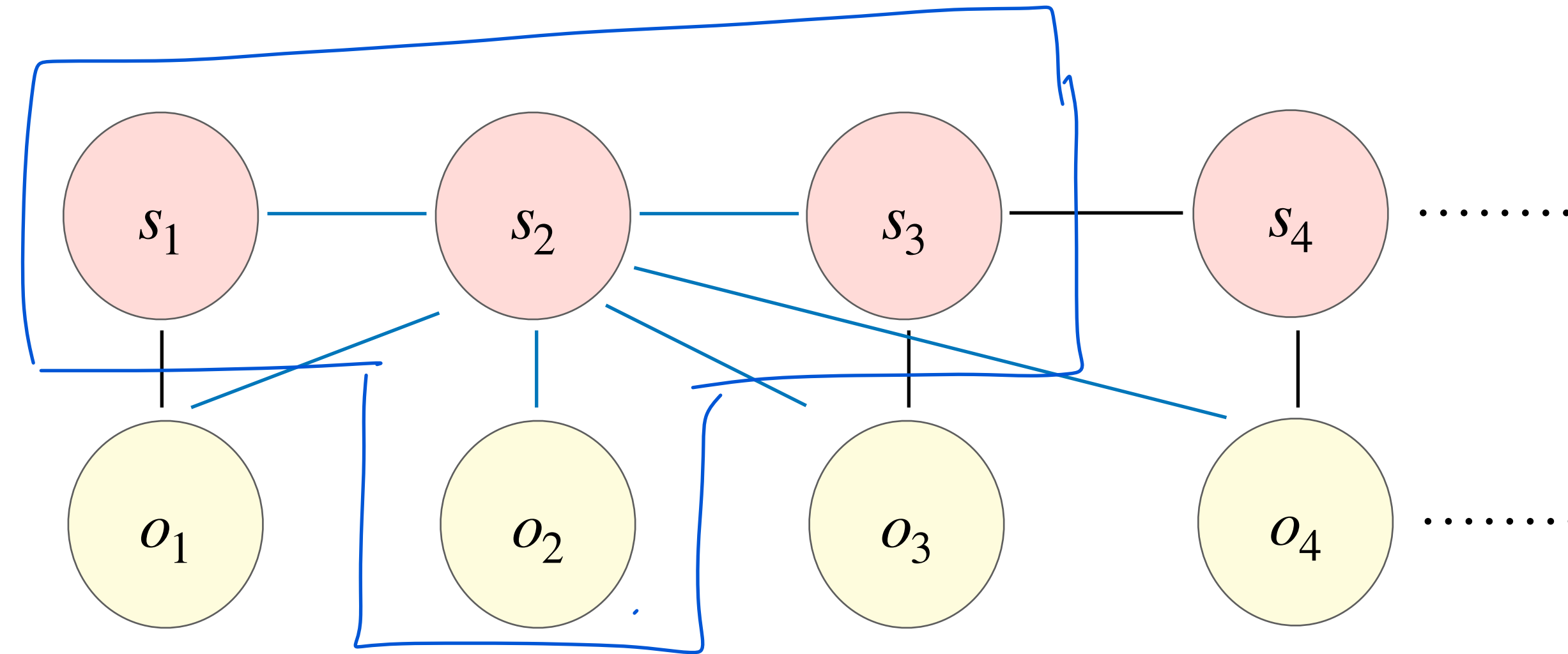
HMM

The/? old/? man/? the/? boat/?

$$\begin{array}{ccccccc}
 P(JJ | DT) & \boxed{P(\text{old} | JJ)} & P(NN | JJ) & \boxed{P(\text{man} | NN)} & P(DT | NN) \\
 P(NN | DT) & \boxed{P(\text{old} | NN)} & P(VB | NN) & \boxed{P(\text{man} | VB)} & P(DT | VB)
 \end{array}$$

Observation bias

Conditional Random Field (advanced)



- Compute log-linear functions over cliques
- Lesser independence assumptions
- Ex: $P(s_t | \text{everything else}) \propto \exp(w \cdot f(s_{t-1}, s_t, s_{t+1}, O))$