Part-of-Speech Tagging and Hidden Markov Model

Bonan Min

bonanmin@gmail.com

Some slides are based on class materials from Thien Huu Nguyen and Ralph Grishman

Parts of Speech (POS)

Role of parts-of-speech in grammar

- ‘preterminals’
- Rules stated in terms of classes of words sharing syntactic properties

noun

verb

adjective

…

Parts of Speech (POS)

The distributional hypothesis: Words that appear in similar contexts have similar representations (and similar meanings)

Substitution test for POS: if a word is replaced by another word, does the sentence remain grammatical?

He noticed the elephant before anybody else

dog

cat

point

features

**what*

**and*

Substitution Test

These can often be too strict; some contexts admit substitutability for some pairs but not others.

He noticed the elephant before anybody else

**Sandy*

Both nouns
but common vs. proper

Both verbs
but transitive vs. intransitive

*He *arrived the elephant before*

Parts of Speech (POS)

Nouns	People, places, things, actions-made-nouns ("I like swimming"). Inflected for singular/plural
Verbs	Actions, processes. Inflected for tense, aspect, number, person
Adjectives	Properties, qualities. Usually modify nouns
Adverbs	Qualify the manner of verbs ("She ran downhill extremely quickly yesterday")
Determiner	Mark the beginning of a noun phrase ("a dog")
Pronouns	Refer to a noun phrase (he, she, it)
Prepositions	Indicate spatial/temporal relationships (on the table)
Conjunctions	Conjoin two phrases, clauses, sentences (and, or)

POS Tag Sets (Categories)

Most influential tag sets were those defined for projects to produce large POS-annotated corpora:

Brown corpus

- 1 million words from variety of genres
- 87 tags

UPenn Tree Bank

- initially 1 million words of Wall Street Journal
- later retagged Brown
- first POS tags, then full parses
- 45 tags (some distinctions captured in parses)

Cross-lingual considerations for POS tags

Penn Treebank POS Tags

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*			

Penn Treebank POS Tags

Verbs

Tag	Description	Examples
VB	base form (found in imperatives, infinities and subjunctives)	• Just do it • You should do it • He wants to do it
VBD	past tense	• He ate the food
VBG	present participle (Verb forms in the gerund or present participle; generally end in -ing)	• He was going to the store • She is implementing the algorithm
VBN	past participle	• The apple was eaten • He had expected to go
VBP	present (non 3rd-sing)	• I am the food • You are tall • We are tall • They do the job
VBZ	present (3rd-sing)	• She is tall • He likes ice cream
MD	modal verbs (All verbs that don't take -s ending in third-person singular present)	• can, could, dare, may, might, must, ought, shall, should, will, would

```
4057 will/md
2973 would/md
1483 could/md
1233 can/md
1066 may/md
 598 should/md
 459 might/md
 332 must/md
 326 wo/md
 246 ca/md
```

http://www.personal.psu.edu/faculty/x/x/xxl13/teaching/sp07/apling597e/resources/Tagset.pdf

Nouns

Tag	Description	Examples
NN	non-proper, singular or mass	the company
NNS	non-proper, plural	the companies
NNP	proper, singular	Carolina
NNPS	proper, plural	Carolinas

RP (Particle)

Used in combination with a verb

- She turned the paper over

verb + particle = phrasal verb, often non-compositional

- turn down, rule out, find out, go on

```
774 up/rp
487 out/rp
301 off/rp
209 down/rp
124 in/rp
 98 over/rp
 81 on/rp
 72 back/rp
 46 around/rp
 25 away/rp
```

DT and PDT

DT (Articles)

- Articles (a, the, every, no)
- Indefinite determiners (another, any, some, each)
- That, these, this, those when preceding noun
- All, both when not preceding another determiner or possessive pronoun

```
65548 the/dt
26970 a/dt
 4405 an/dt
 3115 this/dt
 2117 some/dt
 2102 that/dt
 1274 all/dt
 1085 any/dt
  953 no/dt
  778 those/dt
```

PDT (Predeterminer)

- Determiner-like words that precede an article or possessive pronoun
  - all his marbles
  - both the girls
  - such a good time

```
263 all/pdt
114 such/pdt
 84 half/pdt
 24 both/pdt
  7 quite/pdt
  2 many/pdt
  1 nary/pdt
```

PRP and PRP$

PRP (personal pronoun)

- Personal pronouns (I, me, you, he, him, it, etc.)
- Reflective pronouns (ending in -self): himself, herself
- Nominal possessive pronouns: mine, yours, hers

```
 7854 it/prp
4601 he/prp
 3260 they/prp
2323 his/prp$
 1792 we/prp
 1584 i/prp
 1001 you/prp
  874 them/prp
  694 she/prp
 438 him/prp
```

PRP$ (possessive pronouns)

- Adjectival possessive forms: my, their, its, his, her

```
5013 its/prp$
2364 their/prp$
2323 his/prp$
 521 our/prp$
 430 her/prp$
 328 my/prp$
 269 your/prp$
```

Adjectives

```
2002 other/jj
1925 new/jj
1563 last/jj
1174 many/jj
1142 such/jj
1058 first/jj
 824 major/jj
 715 federal/jj
 698 next/jj
 644 financial/jj
```

JJ (Adjectives)

- General adjectives (happy person, new house)
- Ordinal numbers (fourth cat)

```
1498 more/jjr
 518 higher/j
 432 lower/jj
 285 less/jjr
 158 better/j
 136 smaller/
 122 earlier/
 112 greater/
  93 larger/j
  75 bigger/j
```

JJR (Comparative adjectives)

- Adjectives with a comparative ending -er and comparative meaning (happier person)
- More and less (when used as adjectives) (more mail)

JJS (Superlative adjectives)

- Adjectives with a superlative ending -est and superlative mean (happiest person)
- Most and least (when used as adjectives) (most mail)

```
695 most/jjs
428 least/jjs
315 largest/jjs
299 latest/jjs
209 biggest/jjs
194 best/jjs
 76 highest/jjs
 63 worst/jjs
 31 lowest/jjs
 30 greatest/jjs
```

Adverbs

RB (Adverbs)
- Most words that end in –ly (highly, heavily)
- Degree words (quite, too, very)
- Negative markers (not, n't, never)

RBR (Comparative adverbs)
- Adverbs with a comparative ending -er and comparative meaning, e.g., run faster
- More/less, e.g., more expensive

RBS (Superlative adverbs)
- Adverbs with a superlative ending -est and superlative meaning, e.g., run fastest
- Most/least, e.g., most expensive

```
4410 n't/rb
2071 also/rb
1858 not/rb
1109 now/rb
1070 only/rb
1027 as/rb
 961 even/rb
 839 so/rb
 810 about/rb
 804 still/rb

1121 more/rbr
 516 earlier/rbr
 192 less/rbr
  88 further/rbr
  82 lower/rbr
  75 better/rbr
  65 higher/rbr
  57 longer/rbr
  53 later/rbr
  34 faster/rbr

 549 most/rbs
  21 best/rbs
   9 least/rbs
   8 hardest/rbs
   2 most/rbs|jjs
   1 worst/rbs
   1 rbs/nnp
   1 highest/rbs
   1 earliest/rbs
```

IN and CC

IN (preposition, subordinating conjunction)

- All prepositions (except to) and subordinating conjunctions
  - He jumped on the table because he was excited

31111	of/in
22967	in/in
11425	for/in
7181	on/in
6684	that/in
6399	at/in
6229	by/in
5940	from/in
5874	with/in
5239	as/in

CC (coordinating conjunction)

- And, but, not, or
- Math operators (plus, minor, less, times)
- For (meaning "because")
  - he asked to be transferred, for he was unhappy

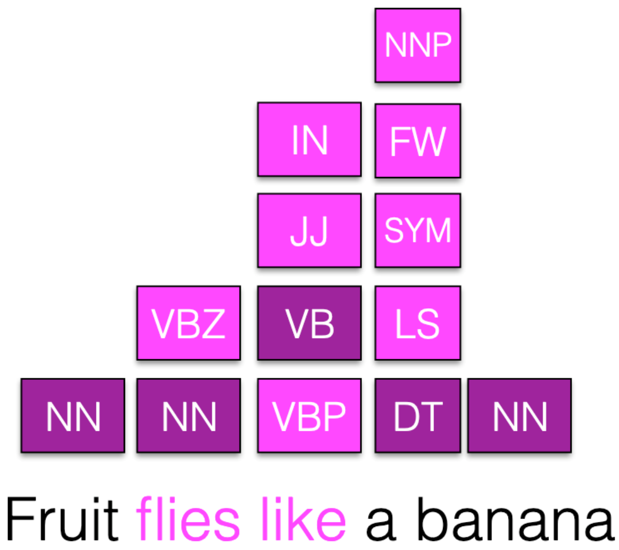
22362	and/cc
4604	but/cc
3436	or/cc
1410	&/cc
94	nor/cc
68	either/cc
53	yet/cc
53	plus/cc
37	both/cc
32	neither/cc

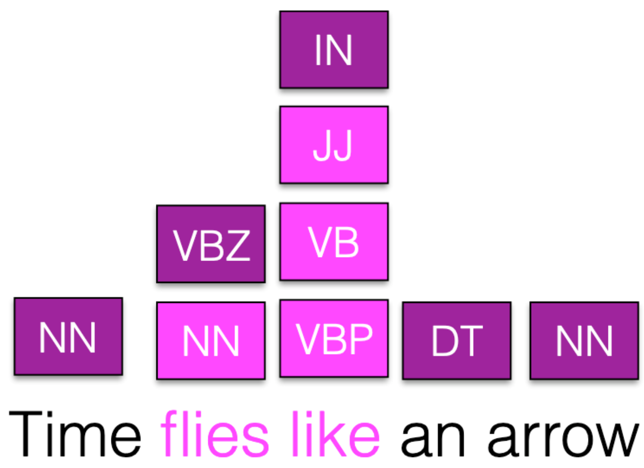
The POS Tagging Task

Task: assigning a POS to each word

not trivial: many words have several tags

dictionary only lists possible POS, independent of context



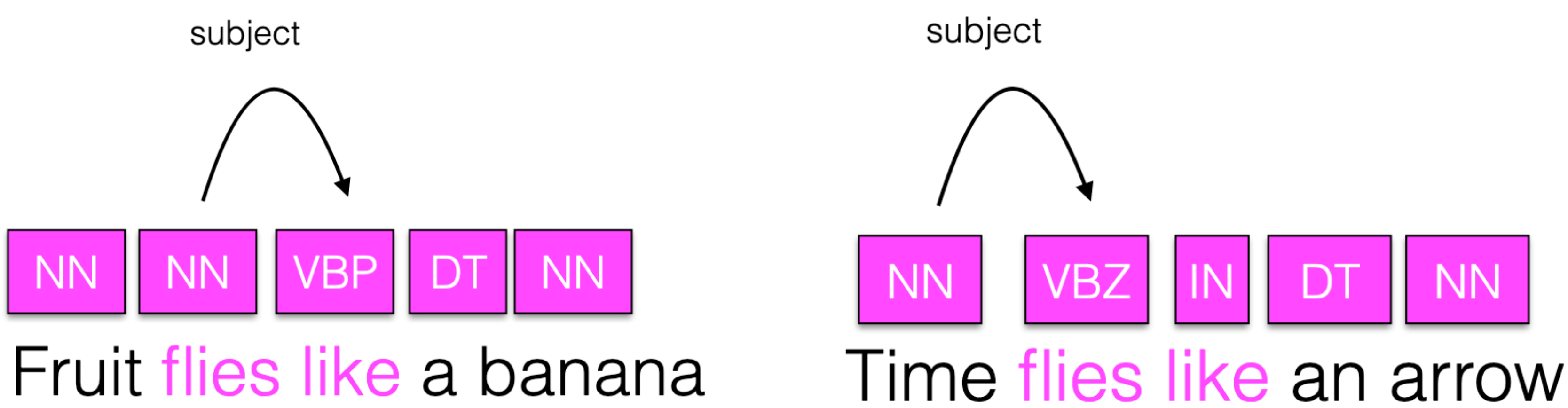


Why Tag?

POS tagging can help parsing by reducing ambiguity

Can resolve some pronunciation ambiguities for text-to-speech (“desert” – noun: /ˈdɛzərt/, verb: /dɪˈzɜrt/)

Can resolve some semantic ambiguities



Some Tricky Cases

JJ or VBN

- If it is gradable (can insert "very") = JJ
  - He was very surprised — JJ
- If can be followed by a "by" phrase = VBN. If that conflicts with #1 above, then = JJ
  - He was invited by some friends of her — VBN
  - He was very surprised by her remarks — JJ

JJ or NNP/NNPS

- Proper names can be adjectives or nouns
  - French cuisine is delicious — JJ
  - The French tend to be inspired cooks — NNPS

Some Tricky Cases

NN or VBG

- Only nouns can be modified by adjectives; only gerunds(-ing) can be modified by adverbs
  - Good cooking is something to enjoy
  - Cooking well is a useful skill

NN

VBG

IN or RP

- If it can precede or follow the noun phrase = RP
  - She told off her friends
  - She told her friends off
- If it must precede the noun phrase = IN
  - She stepped off the train
  - *She stepped the train off

Quiz [SLP2]

Find the tagging errors in the following sentences:

I/PRP need/VBP a/DT flight/NN from/IN Atlanta/NN

Does/VBZ this/DT flight/NN serve/VB dinner/NNS

I/PRP have/VB a/DT friend/NN living/VBG in/IN Denver/NNP

Can/VBP you/PRP list/VB the/DT nonstop/JJ afternoon/NN flights/NNS

Quiz [SLP2]

Find the tagging errors in the following sentences:

I/PRP need/VBP a/DT flight/NN from/IN Atlanta/NN
NNP

Does/VBZ this/DT flight/NN serve/VB dinner/NNS
NN

I/PRP have/VB a/DT friend/NN living/VBG in/IN Denver/NNP
VBP

Can/VBP you/PRP list/VB the/DT nonstop/JJ afternoon/NN flights/NNS
MD

POS Tagging Methods

Similar to text classification, we would like to use machine learning methods to do POS tagging.

Using supervised learning, we need to assemble a text corpus and manually annotate the POS for every word in the corpus (i.e., the Brown corpus) (i.e., the corpus-based methods).

- We can divide the corpus into training data, development data and test data

To build a good corpus

- we must define a task people can do reliably (choose a suitable POS set)
- we must provide good documentation for the task
  - so annotation can be done consistently
- we must measure human performance (through dual annotation and inter-annotator agreement)
- Often requires several iterations of refinement

The Simplest POS Tagging Method

We tag each word with its most likely part-of-speech (based on the training data)

- this works quite well: about 90% accuracy when trained and tested on similar texts
- although many words have multiple parts of speech, one POS typically dominates within a single text type

How can we take advantage of context to do better?

POS Tagger As Sequence Labeling

Sequence labeling: given a sequence of observations $x = x_1, x_2, \dots, x_n$, we need to assign a label/type/class y_i for each observation $x_i \in x$, leading to the sequence label $y = y_1, y_2, \dots, y_n$ for x ($y_i \in Y$) (Y is the set of possible POS tags)

For POS tagging, x can be an input sentence where x_i is the i-th word in the sentence, and y_i can be the POS tag of x_i in x (Y is the set of the possible POS tags in our data). E.g.,

x =	Does	this	flight	serve	dinner
y =	VBZ	DT	NN	VB	NN

Sequence Labeling

As in text classification, we also want to estimate the distribution from the training data:

$$P(y|x) = P(y_1, y_2, \dots, y_n | x_1, x_2, \dots, x_n)$$

So, we can also obtain the predicted label sequence for x by:

$$y^* = argmax_y P(y|x) = argmax_y P(y_1, y_2, \dots, y_n | x_1, x_2, \dots, x_n)$$

Hidden Markov Model (HMM)

Using Bayes' Rule

$$argmax_y P(y|x) = argmax_y \frac{P(x|y)P(y)}{P(x)}$$

$$= argmax_y P(x|y)P(y)$$

$$= argmax_c P(x_1, x_2, \dots, x_n | y_1, y_2, \dots, y_n) P(y_1, y_2, \dots, y_n)$$

First-order Markov assumption: the probability of the label for the current step only depends on the label from the previous step, so:

$$P(y_1, y_2, \dots, y_n) = \prod_{t=1}^{n} P(y_t | y_{<t}) = \prod_{t=1}^{n} P(y_t | y_{t-1})$$

Independency assumption: the probability of the current word is only dependent on its label:

$$P(x_1, x_2, \dots, x_n | y_1, y_2, \dots, y_n) = \prod_{t=1}^{n} P(x_t | x_{<t}, y) = \prod_{t=1}^{n} P(x_t | y_t)$$

So, in HMM, we need to obtain two types of probabilities:

- The transition probabilities: $P(y_t | y_{t-1})$
- The emission probabilities: $P(x_t | y_t)$

Parameter Estimation

Using Maximum Likelihood Estimators as in Naïve Bayes (i.e., just counting):

How many times y_{t-1} and y_t appear together in the training data?

$$P(y_t|y_{t-1}) = \frac{c(y_{t-1},y_t)}{c(y_{t-1})}$$

How many times y_{t-1} appears in the training data?

$$P(x_t|y_t) = \frac{c(x_t,y_t)}{c(y_t)}$$

How many times x_t appears with y_t in the training data?

With smoothing:

$$P(x_t|y_t) = \frac{\alpha + c(x_t,y_t)}{|Y|\alpha + c(y_t)}$$

How many probabilities we have?

Transition Probabilities

	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

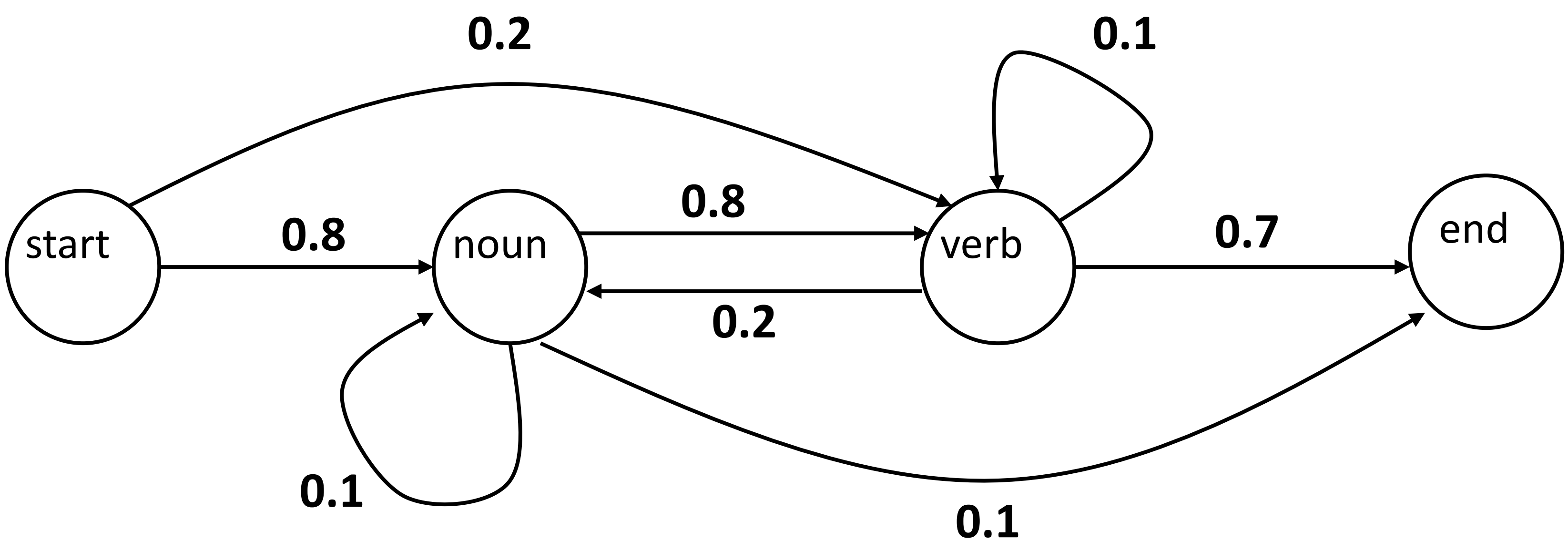
Figure 10.5 The A transition probabilities $P(t_i|t_{i-1})$ computed from the WSJ corpus without smoothing. Rows are labeled with the conditioning event; thus $P(VB|MD)$ is 0.7968.

Emission Probabilities

	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

Figure 10.6 Observation likelihoods *B* computed from the WSJ corpus without smoothing.

Hidden State Network



Decoding

Given the transition and emission probabilities $P(y_t|y_{t-1})$ and $P(x_t|y_t)$, we need to find the best label sequence $y^* = y_1^*, y_2^*, \ldots, y_n^*$ for the input sentence $x = x_1, x_2, \ldots, x_n$ via:

$$
\begin{aligned}
y^* &= argmax_y P(y|x) \\
&= argmax_y \frac{P(x,y)}{P(x)} = argmax_y P(x,y) \\
&= argmax_y P(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n)
\end{aligned}
$$

This requires the enumeration over all the possible label sequences (paths) y which are exponentially large

- E.g., using Penn Treebank with 45 tags
  - A sentence of length 5 would have 45^5 = 184,528,15 possible sequences
  - A sentence of length 20 would have 45^{20} = 1.16e33 possible sequences

Greedy Decoder

simplest decoder (tagger) assign tags deterministically from left to right

selects y_t^* to maximize $P(x_t|y_t) \; * P(y_t|y_{t-1})$

does not take advantage of right context

can we do better?

Viterbi Algorithm

Basic idea: if an optimal path through a sequence uses label L at time t, then it must have used an optimal path to get to label L at time t

We can thus discard all non-optimal paths up to label L at time t

Let $v_t(s)$ be the probability that the HMM is in state (label) s after seeing the first t observations (words) and passing through the most probable state sequence $y_1, y_2, \dots, y_{t-1}$:

$$v_t(s) = max_{y_1, y_2, \dots, y_{t-1}} P(x_1, x_2, \dots, x_t, y_1, y_2, \dots, y_{t-1}, y_t = s)$$

Introducing the $start$ and end states to represent the beginning and the end of the sentences ($y_0 = start, y_{n+1} = end$), the probability for the optimal label sequence would be:

$$v_{n+1}(end) = max_{y_1, y_2, \dots, y_n} P(x_1, x_2, \dots, x_n, y_0 = start, y_1, y_2, \dots, y_n, y_{n+1} = end)$$

Viterbi Algorithm

$v_t(s) = \ max_{y_1,y_2,\dots,y_{t-1}} P(x_1, x_2, \dots, x_t, y_0 = start, y_1, y_2, \dots, y_{t-1}, y_t = s)$

Initialization ($t = 0$):

$$v_0(s) = \begin{cases} 1 \ if \ s = start \\ 0 \ otherwise \end{cases}$$

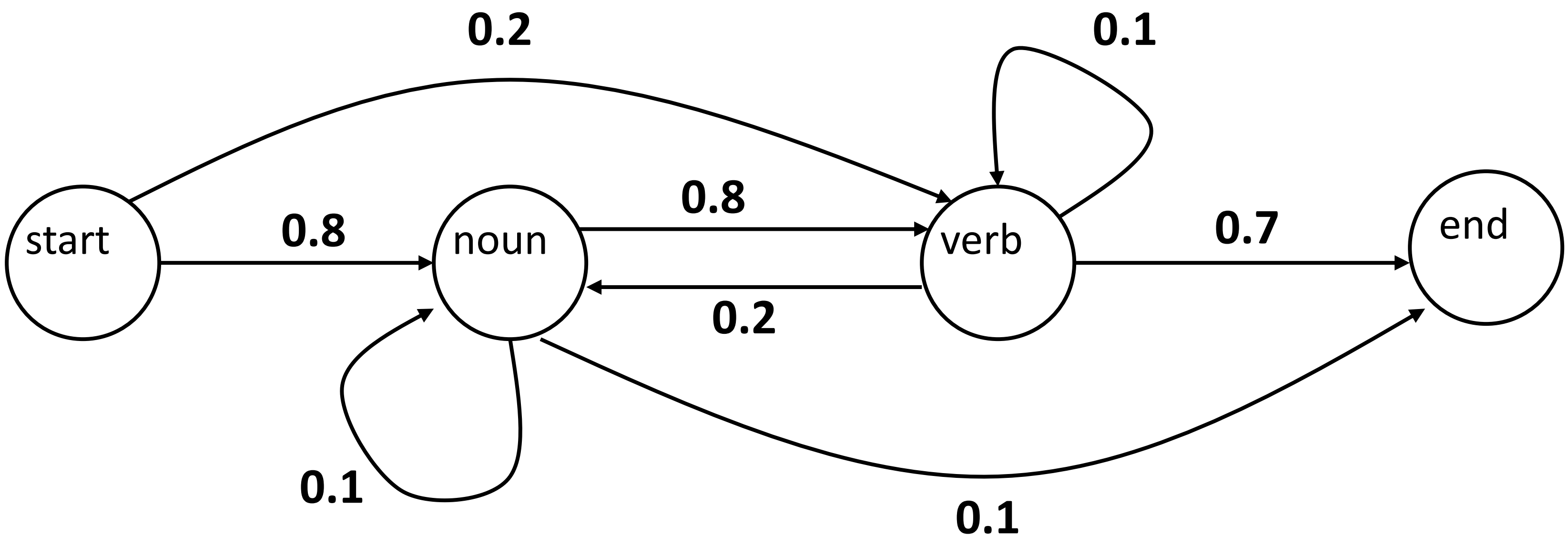
Recurrence ($t > 0$):

$$v_t(s) = max_{s' \in Y}[v_{t-1}(s') P(s|s') P(x_t|s)]$$
$$backtrack_t(s) = argmax_{s' \in Y}[v_{t-1}(s') P(s|s') P(x_t|s)]$$

Termination ($t = n + 1$): the optimal probability is $v_{n+1}(end)$, following the backtrack links (starting at $backtrack_{n+1}(end)$) to retrieve the optimal path.

Example

Fish sleep



Word Emission Probabilities

Word Emission Probabilities P (word | state)

A two-word language: "fish" and "sleep"

Suppose in our training corpus,

- "fish" appears 8 times as a noun and 5 times as a verb
- "sleep" appears twice as a noun and 5 times as a verb

Emission probabilities:

- Noun
  - P(fish | noun) : 0.8
  - P(sleep | noun) : 0.2
- Verb
  - P(fish | verb) : 0.5
  - P(sleep | verb) : 0.5

Viterbi Probabilities

	0	1	2	3
start				
verb				
noun				
end				

Noun

P(fish | noun) : 0.8

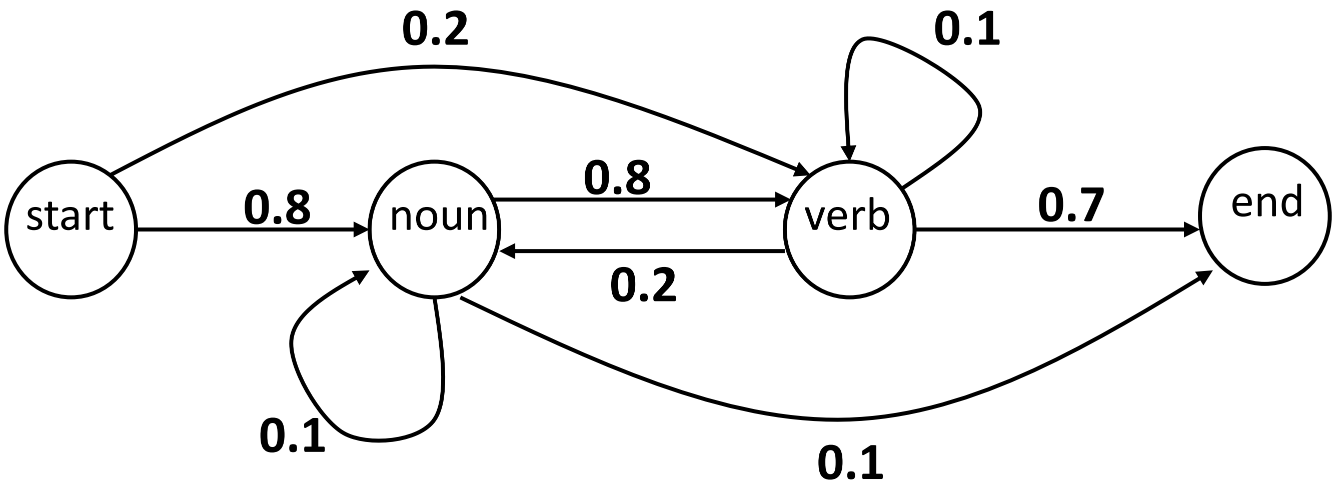
P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5

Init



	0	1	2	3
start	1			
verb	0			
noun	0			
end	0			

Noun

P(fish | noun) : 0.8

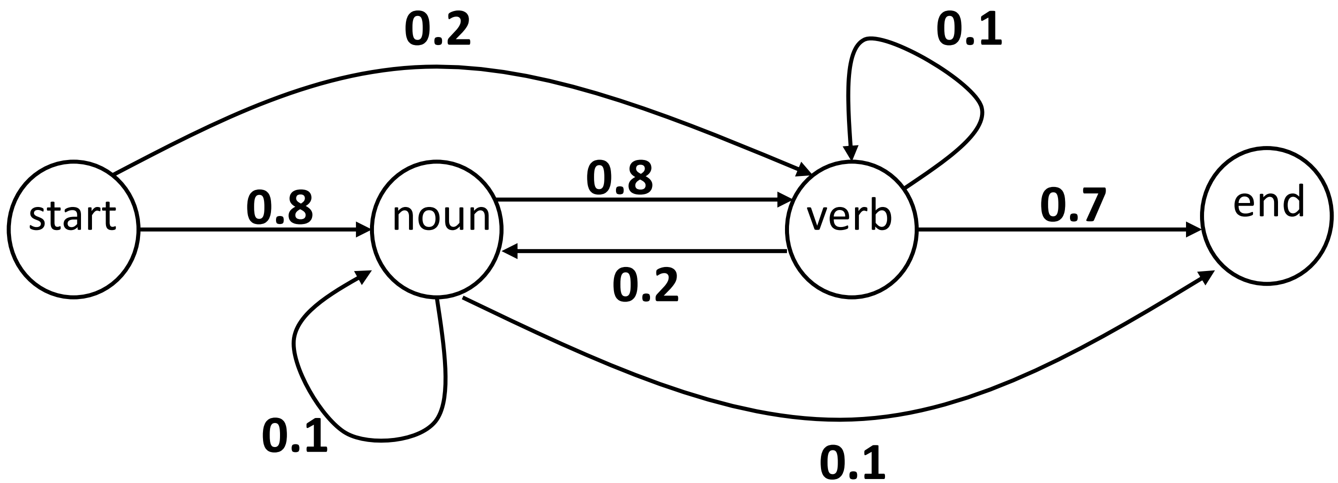
P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5

Token 1: fish



	0	1	2	3
start	1	0		
verb	0	.2 * .5		
noun	0	.8 * .8		
end	0	0		

Noun

P(fish | noun) : 0.8

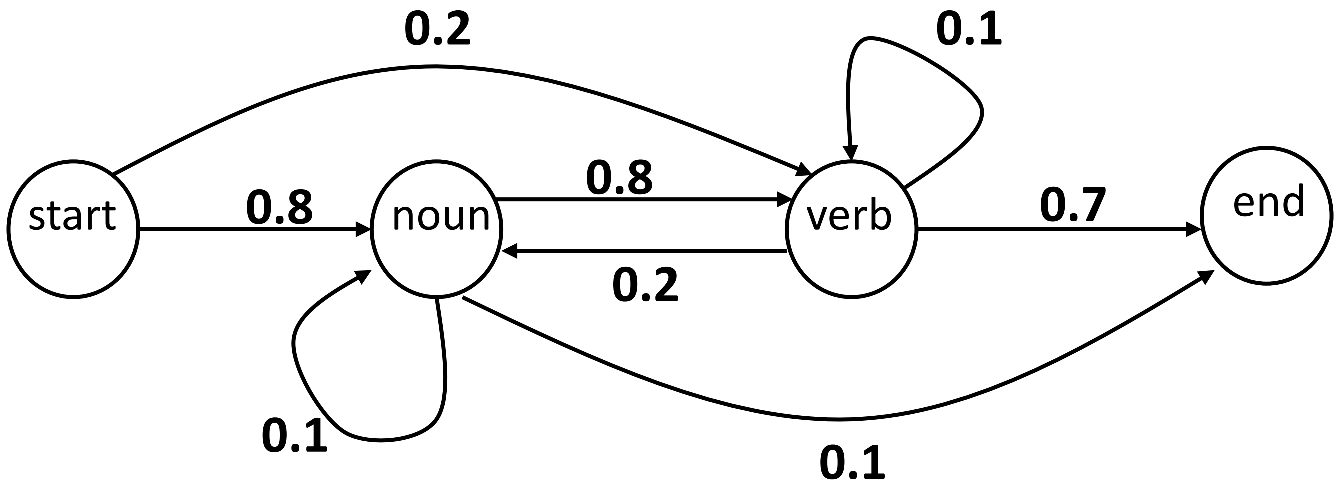
P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5

Token 1: fish



	0	1	2	3
start	1	0		
verb	0	.1		
noun	0	.64		
end	0	0		

Noun

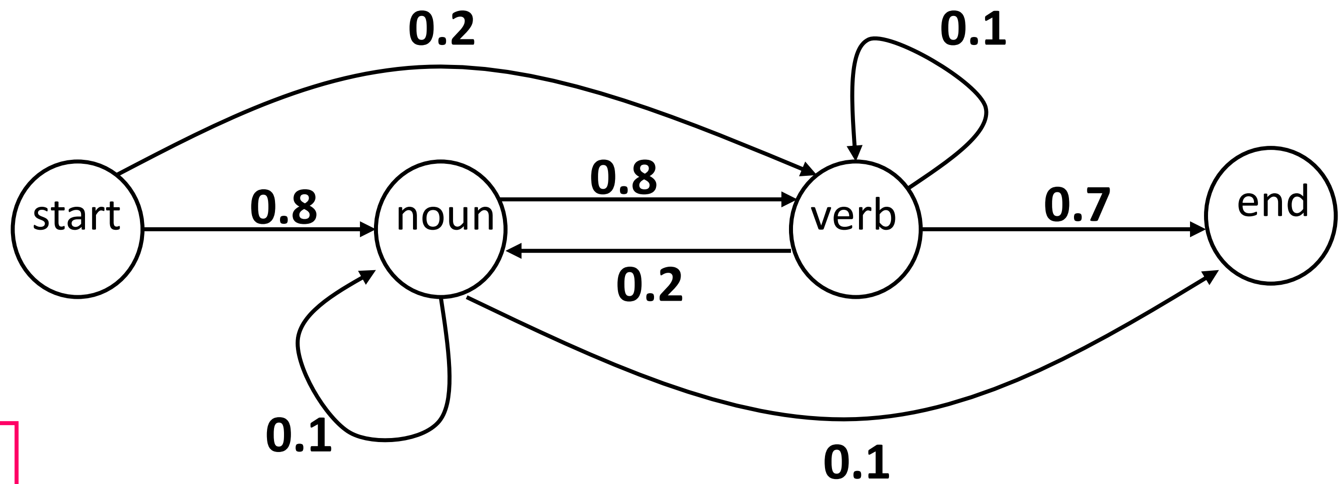
P(fish | noun) : 0.8

P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5



Token 2: sleep

(if 'fish' is verb)

	0	1	2	3
start	1	0	0	
verb	0	.1	.1*.1*.5	
noun	0	.64	.1*.2*.2	
end	0	0	-	

Noun

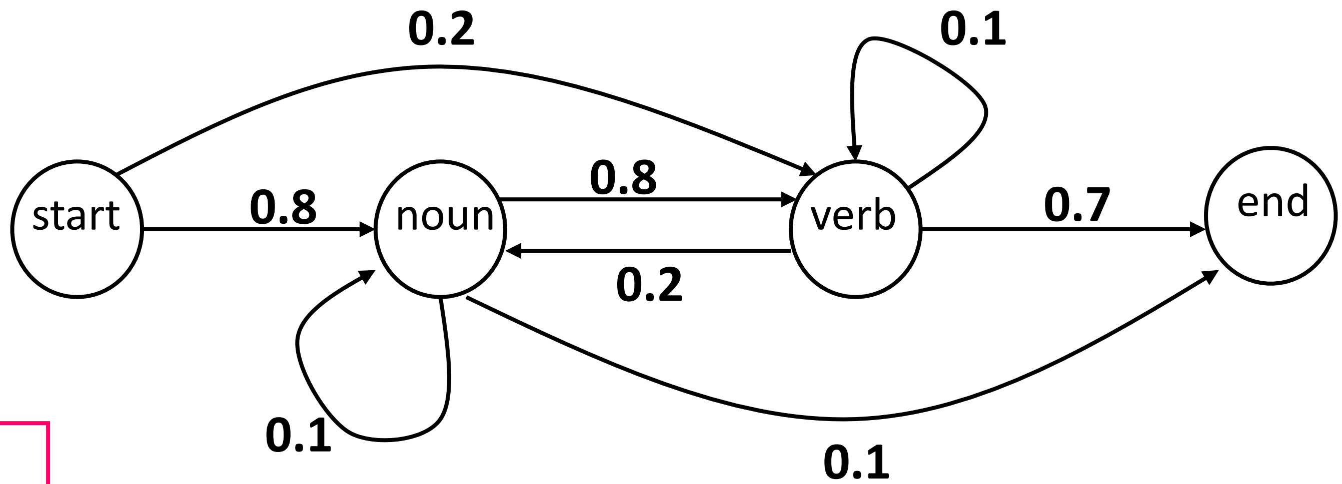
P(fish | noun) : 0.8

P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5



Token 2: sleep

(if 'fish' is a noun)

	0	1	2	3
start	1	0	0	
verb	0	.1	.005 .64*.8*.5	
noun	0	.64	.004 .64*.1*.2	
end	0	0	-	

Noun

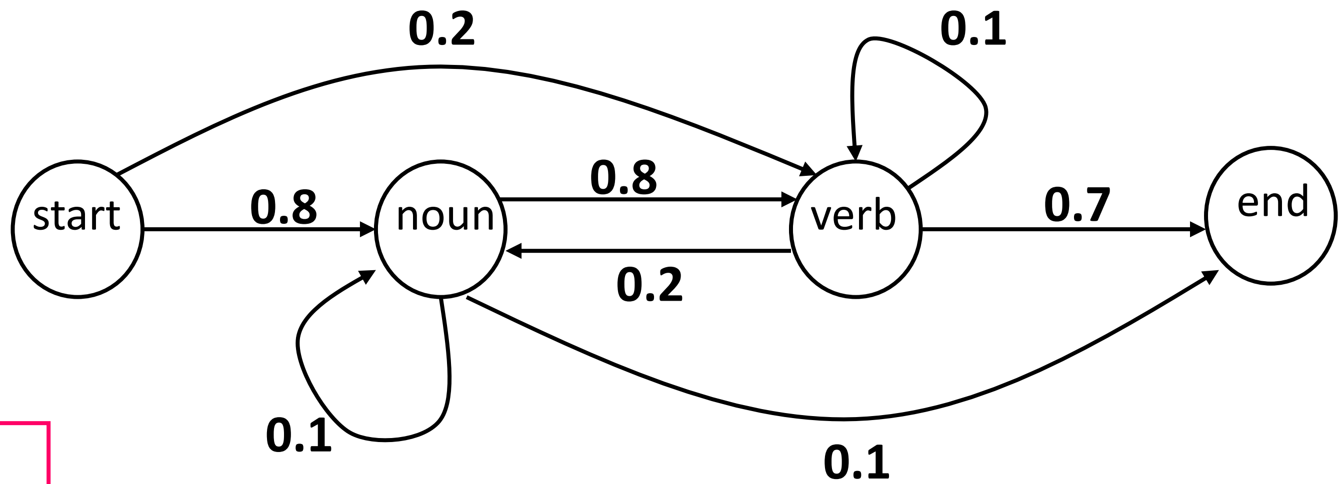
P(fish | noun) : 0.8

P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5



Token 2: sleep

(if ‘fish’ is a noun)

	0	1	2	3
start	1	0	0	
verb	0	.1	.005 .256	
noun	0	.64	.004 .0128	
end	0	0	-	

Noun

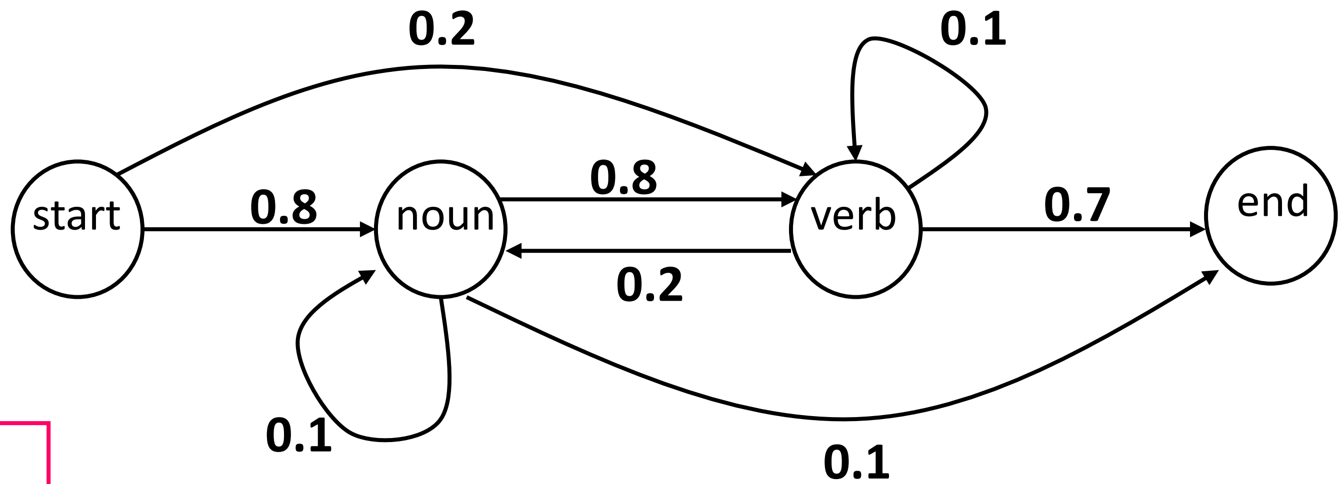
P(fish | noun) : 0.8

P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5



Token 2: sleep
take maximum,
set back pointers

	0	1	2	3
start	1	0	0	
verb	0	.1	~~.005~~ .256	
noun	0	.64	~~.004~~ .0128	
end	0	0	-	

Noun

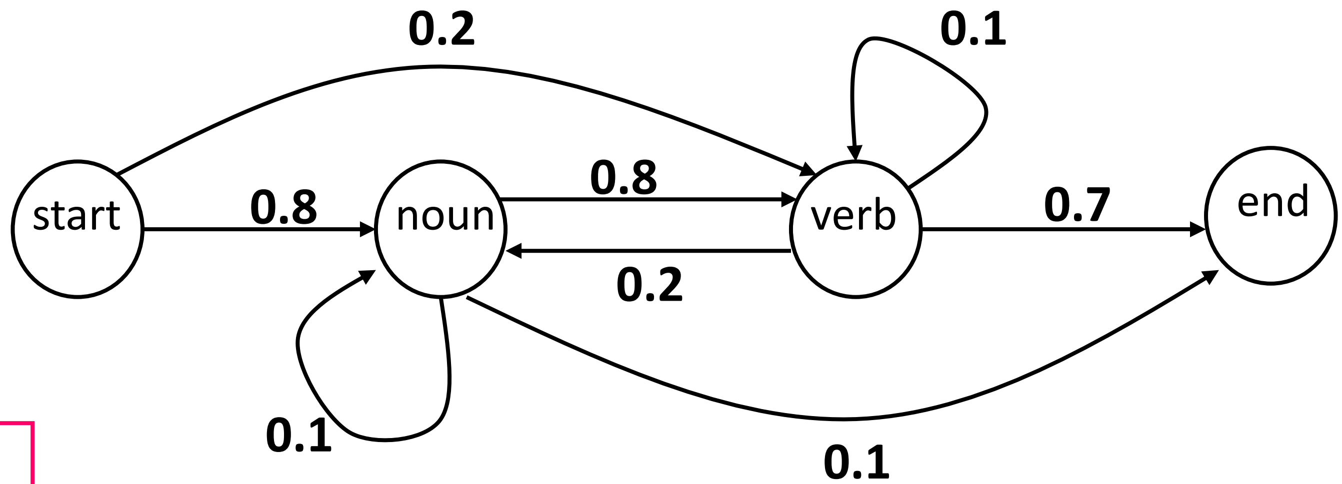
P(fish | noun) : 0.8

P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5



Token 2: sleep
take maximum,
set back pointers

	0	1	2	3
start	1	0	0	
verb	0	.1	.256	
noun	0	.64	.0128	
end	0	0	-	

Noun

P(fish | noun) : 0.8

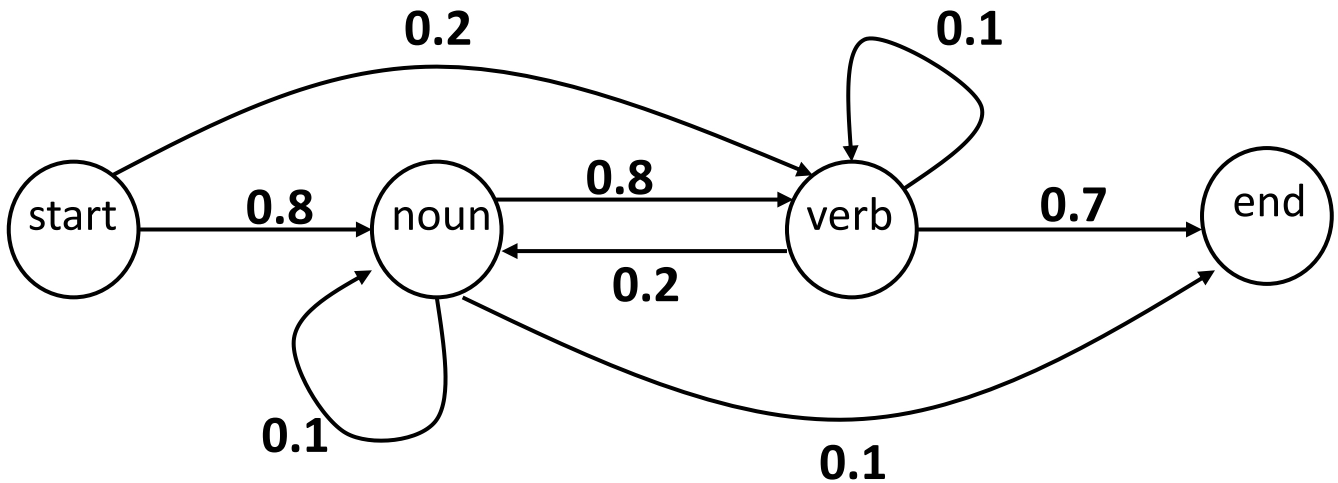
P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5

Token 3: end



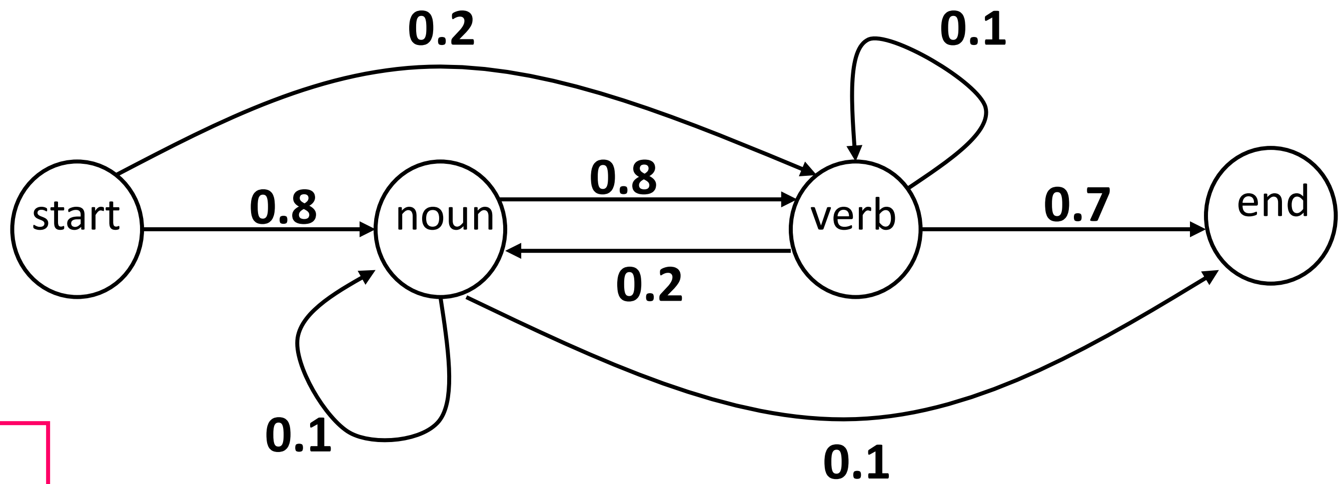
	0	1	2	3
start	1	0	0	0
verb	0	.1	.256	-
noun	0	.64	.0128	-
end	0	0	-	.256*.7 .0128*.1

Noun

- P(fish | noun) : 0.8
- P(sleep | noun) : 0.2

Verb

- P(fish | verb) : 0.5
- P(sleep | verb) : 0.5



Token 3: end
take maximum,
set back pointers

	0	1	2	3
start	1	0	0	0
verb	0	.1	.256	-
noun	0	.64	.0128	-
end	0	0	-	.256*.7 ~~.0128*.1~~

Noun

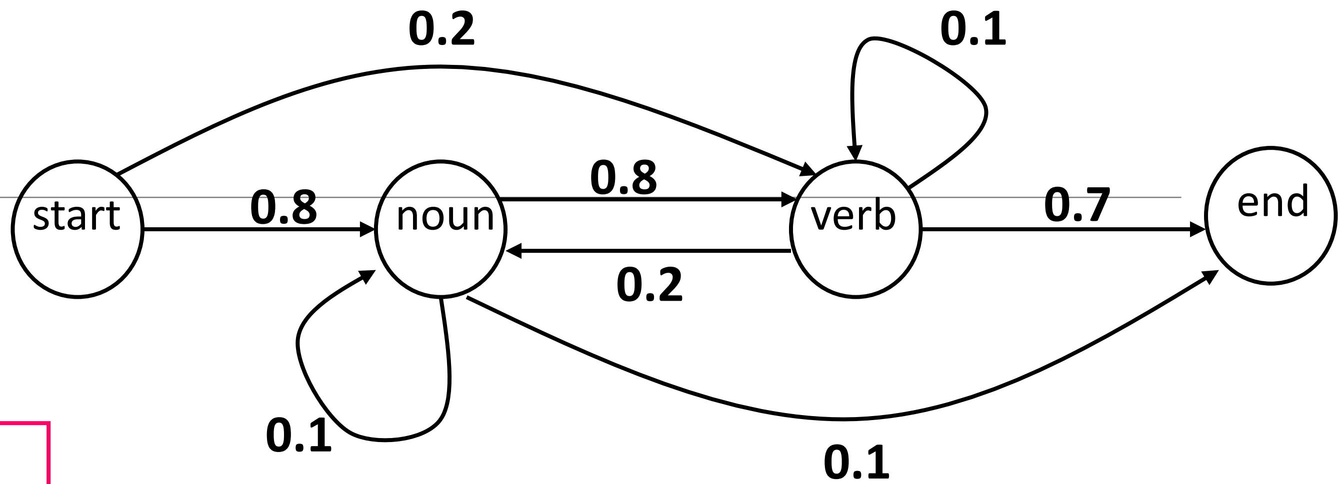
P(fish | noun) : 0.8

P(sleep | noun) : 0.2

Verb

P(fish | verb) : 0.5

P(sleep | verb) : 0.5



Decode:

fish = noun

sleep = verb

	0	1	2	3
start	1	0	0	0
verb	0	.1	.256	-
noun	0	.64	.0128	-
end	0	0	-	.256*.7

Complexity for Viterbi

time = $O\ (\ s^2\ n)$

for s states (labels) and n words

(Relatively fast: for 40 states and 20 words,

32,000 steps)