CSCI 544, lecture 11: Part-of-speech tagging, Hidden Markov models

Ron Artstein

2022-09-27

Administrative notes: deadlines

Written Assignment Peer Grading will be released after checking for errors

Coding Assignment 2 was due today

Coding Assignment 3 due October 11

Project:

Due Date	Task
September 20	Form project teams (52 teams)
September 20–29	**Initial discussion with TA**
October 4	**Project proposal**
November 3	Project status report
Nov 29/Dec 1	Poster presentations (in class)
December 1	Final report
December 3	Self-evaluation and peer grading

Parts of speech

Syntactic category of a word:

- Noun, verb, . . .

May differ between languages, grammatical theories

Open-class	Noun, verb, adjective, . . .
Closed-class	Preposition, conjunction, determiner, . . .

Tagging: in a text, identify the part of speech of each word.

Part-of-speech tagging

Task driven by ambiguity

Somewhat artificial task.

Why would we want to tag words with POS?

Part-of-speech tagging

Task driven by ambiguity

Somewhat artificial task.

Why would we want to tag words with POS?

- Could be helpful for downstream processes.
- Push technology forward.

Time flies like an arrow

Part-of-speech tagging

Task driven by ambiguity

Somewhat artificial task.

Why would we want to tag words with POS?

- Could be helpful for downstream processes.
- Push technology forward.

Time	**flies**	**like**	**an**	**arrow**	
NN	VBZ	IN	DT	NN	(Penn Treebank tags)

Part-of-speech tagging

Task driven by ambiguity

Somewhat artificial task.

Why would we want to tag words with POS?

- Could be helpful for downstream processes.
- Push technology forward.

Time	**flies**	**like**	**an**	**arrow**	
NN	VBZ	IN	DT	NN	(Penn Treebank tags)
NN	NNS	VBP	DT	NN	

Part-of-speech tagging

Task driven by ambiguity

Somewhat artificial task.

Why would we want to tag words with POS?

- Could be helpful for downstream processes.
- Push technology forward.

Time	flies	like	an	arrow	
NN	VBZ	IN	DT	NN	(Penn Treebank tags)
NN	NNS	VBP	DT	NN	
VB	NNS	IN	DT	NN	

Long-distance dependencies

Flying	planes	can	be	dangerous
$\begin{Bmatrix} \text{VBG} \\ \text{JJ} \end{Bmatrix}$	NNS	MD	VB	JJ

Flying	planes	is	dangerous
VBG	NNS	VBZ	JJ

Flying	planes	are	dangerous
JJ	NNS	VBP	JJ

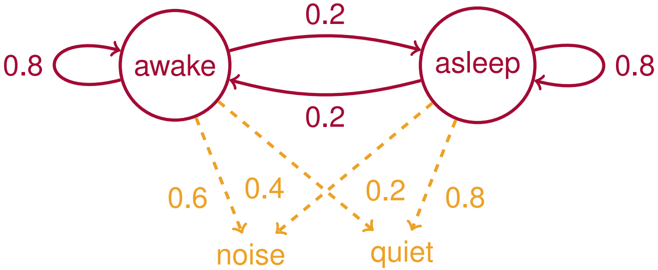
Sequence labeling: each instance may depend on preceding or following labels.

Markov chains

Probabilistic state machine.

- **Transition probabilities:** state $\rightarrow$ state
- **Emission probabilities:** state $\rightarrow$ observation

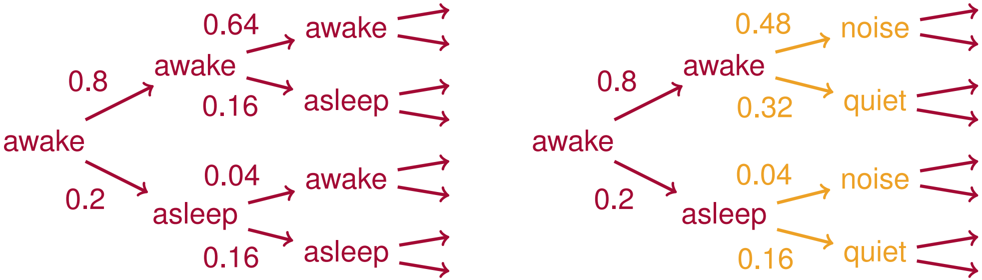
Markov assumption:
Probabilities depend only on state, not on history



	Awake	Asleep
Awake	0.8	0.2
Asleep	0.2	0.8

	Noise	Quiet
Awake	0.6	0.4
Asleep	0.2	0.8

Markov chains are a generative model



Hidden Markov model

Only emissions are observable; states are hidden

- Infer states from observations (most likely sequence)

Decoding (known transition and emission probabilities)

- Viterbi algorithm

Learning probability matrices

- Learn from corpus
- Forward-backward algorithm = special case of EM
  - Expectation: estimate states based on parameters
  - Maximization: estimate parameters based on states

Learning state structure

- Design manually
- Start with fully connected, then learn probabilities

Viterbi decoding example

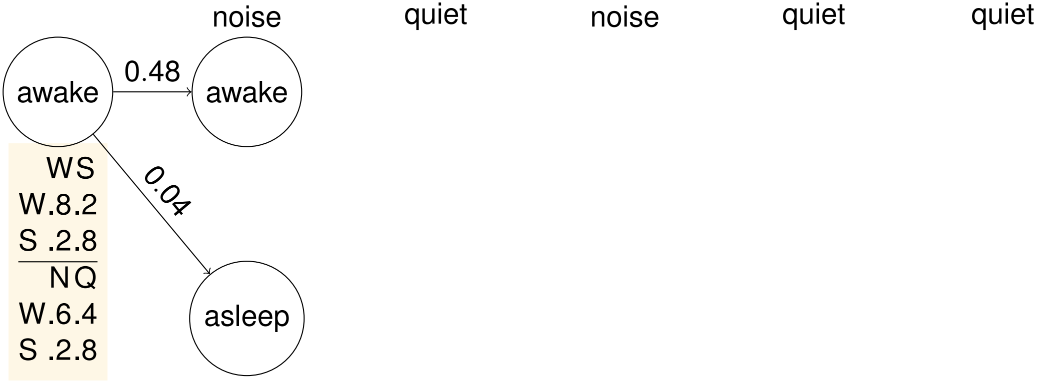
noise quiet noise quiet quiet

awake

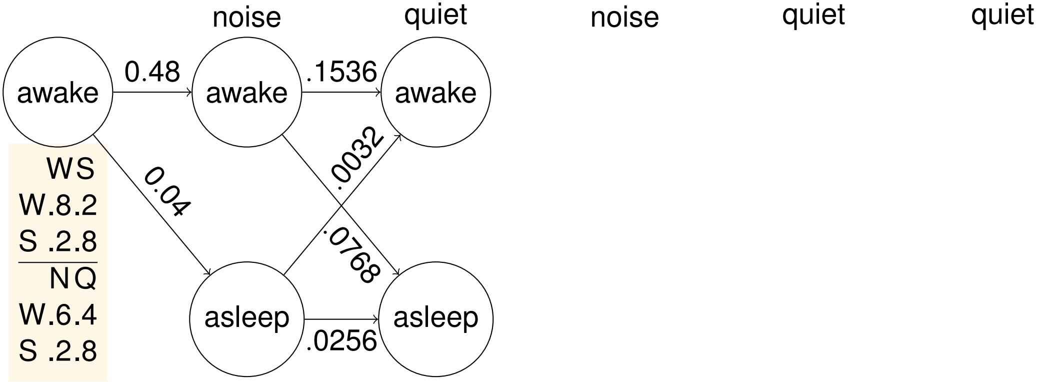
	W	S
W	.8	.2
S	.2	.8

	N	Q
W	.6	.4
S	.2	.8

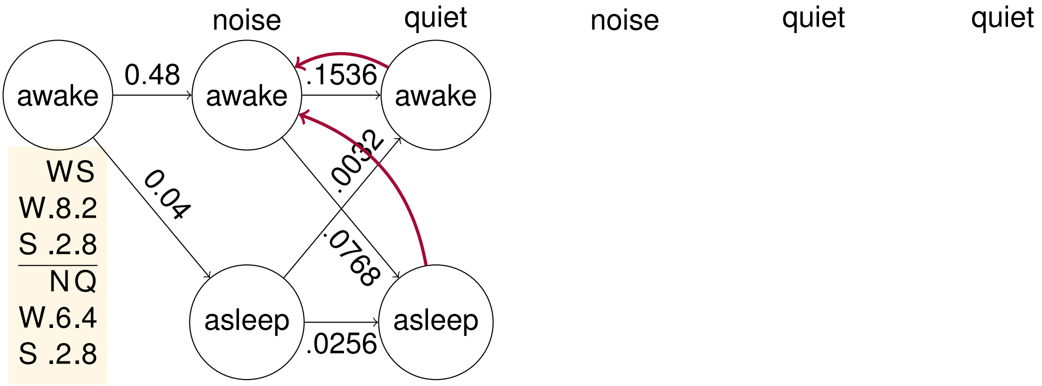
Viterbi decoding example



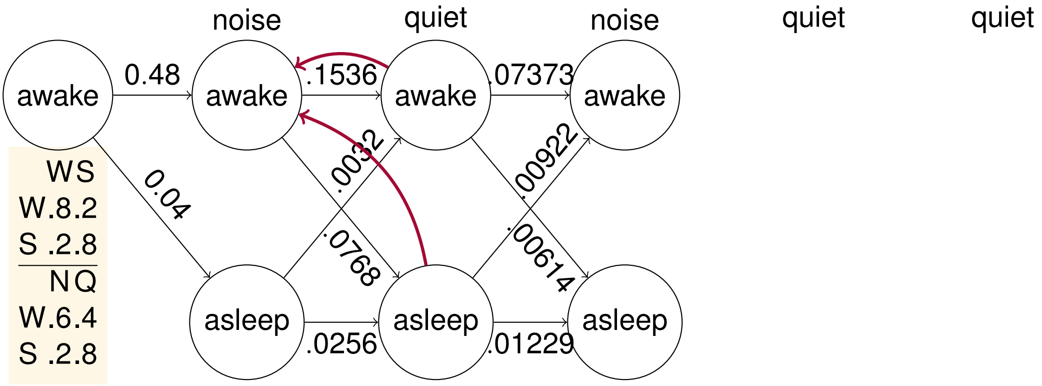
Viterbi decoding example



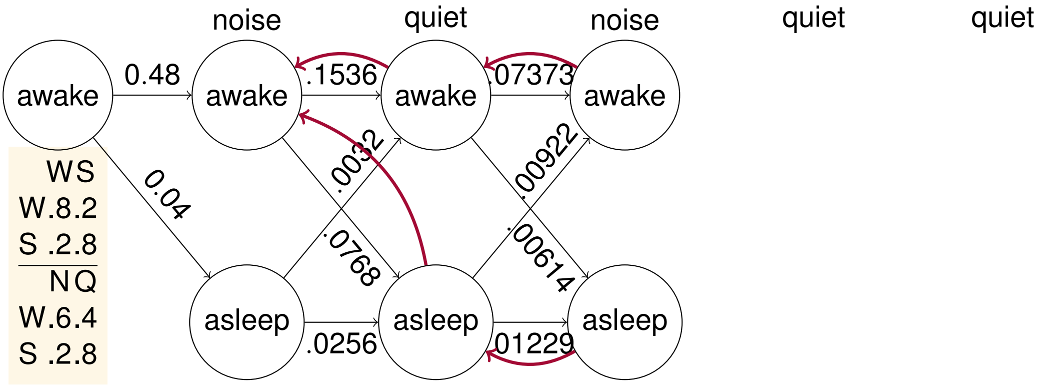
Viterbi decoding example



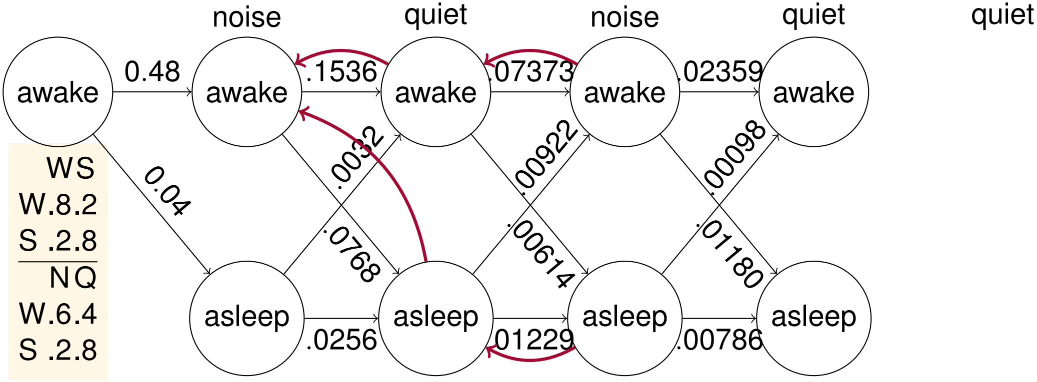
Viterbi decoding example



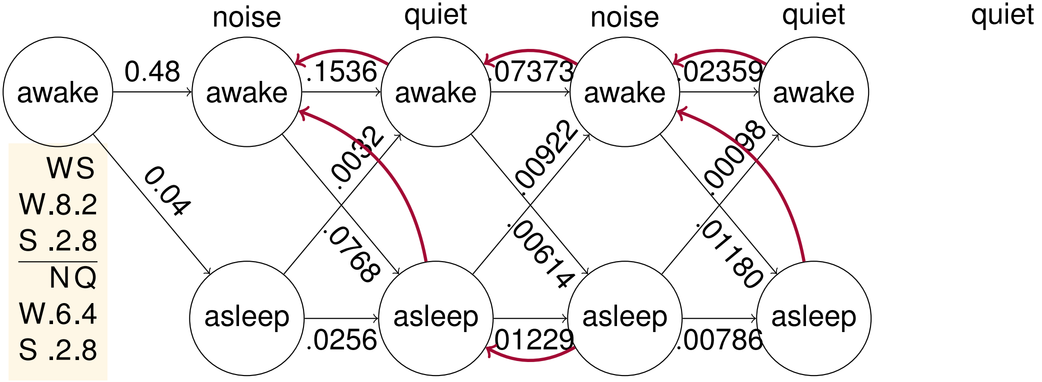
Viterbi decoding example



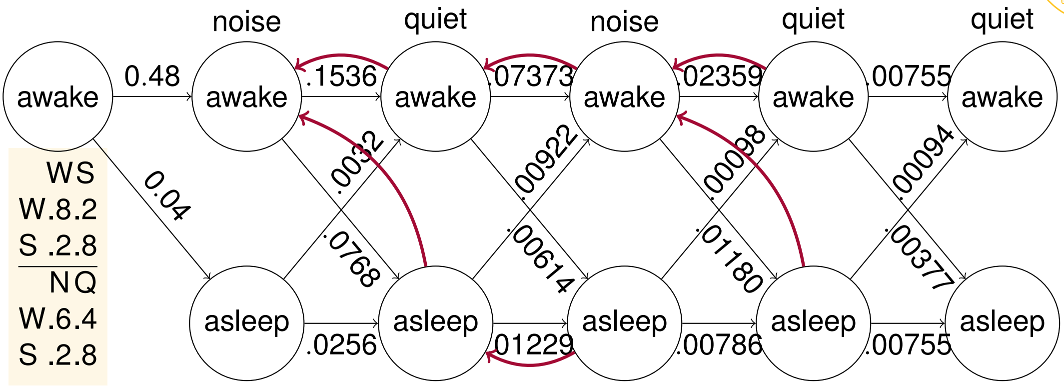
Viterbi decoding example



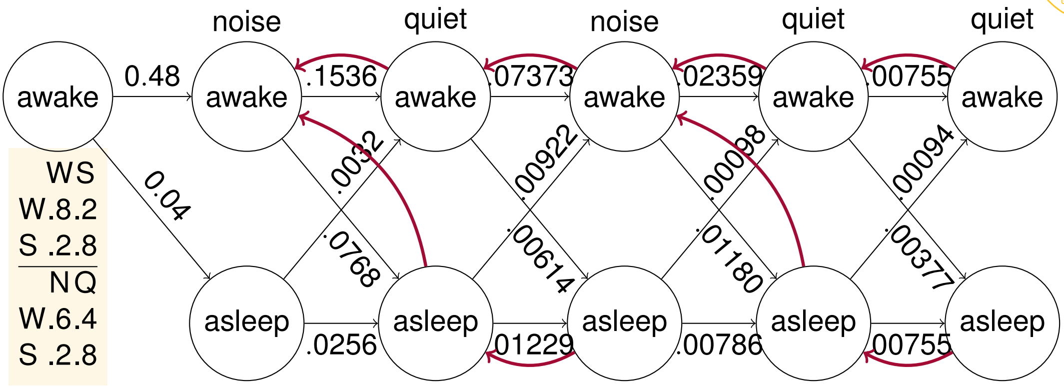
Viterbi decoding example



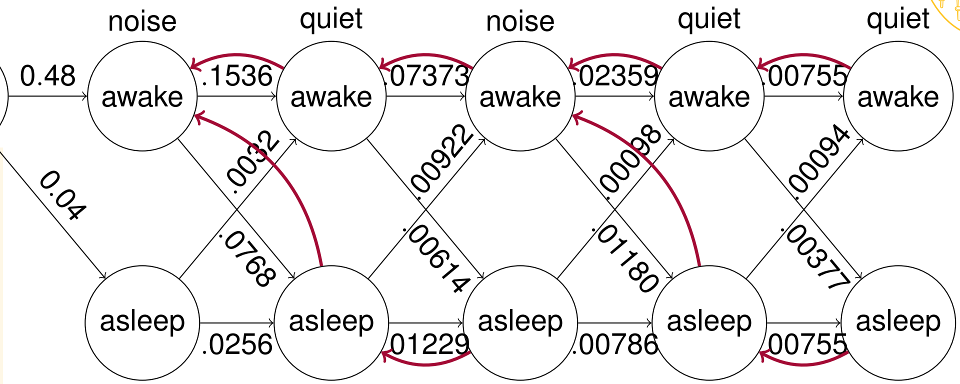
Viterbi decoding example



Viterbi decoding example



Viterbi decoding example



	W	S
W	.8	.2
S	.2	.8

	N	Q
W	.6	.4
S	.2	.8

(previous $\cdot$ transition) $\cdot$ emission $\approx$ prior $\cdot$ conditional

HMMs for POS tagging

Observations = words; what are the states?

- States = POS tags
- Viterbi decoding = infer most probable tag sequence

Learn parameters from corpus

- Emission matrix = tag$\rightarrow$word probabilities $P(\text{word}|\text{current tag})$
- Transition matrix = tag bigram probabilities $P(\text{tag}|\text{previous tag})$

Issues with decoding

- How to handle unknown words?
- Is smoothing useful?

More context can be given with richer states (e.g., tag bigrams)

Coding Assignment 3

Write a Hidden Markov Model part-of-speech tagger

From scratch!

Two langauges for training and development

- Test on unseen data in same langauges
- Test on surprise language

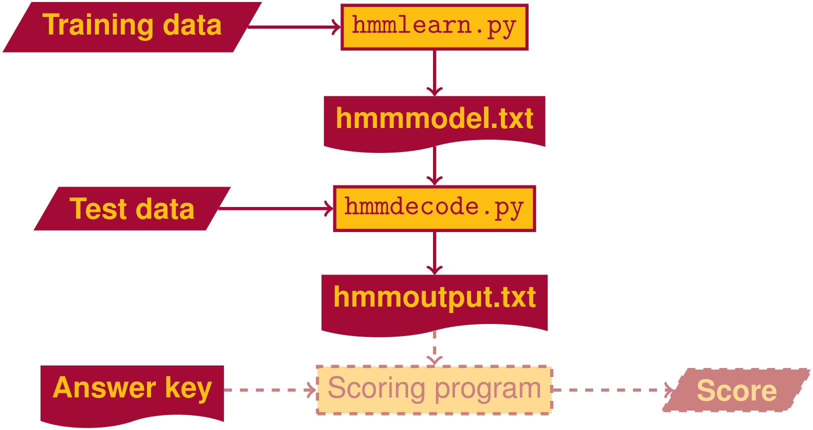
Graded on performance

Programming in Python

Submit on Vocareum

- Automatic feedback
- Submit early, submit often!

Coding assignment 3: programs



Coding assignment 3: notes

Word/tag format

- Learn tags from training data (surprise language!)
- Careful with slash character

Unseen words and transitions

- Unseen words do not have known emissions
- Unseen transitions can cause program to halt
- Reference uses smoothing on transitions, **not** emissions

Runtime efficiency

	Python 2	Python 3
`if item in dict:`	**Fast**	**Fast**
`if item in dict.keys():`	**Slow**	**Fast**
- Don't multiply zeros

Coding assignment 3: decoding efficiency

Probability of state *q* at time *t*

$$P(s,t) = \max_{s' \in State} P(s', t-1) * P(tr(s', s)) * P(em(s, o_t))$$

When $P(em(s, o_t)) = 0$, product doesn't depend on s'!

Since most emissions are 0, an if-statement cuts 90% of runtime

```
if 0 == em[q][obs[t]]:
    prob[q][t] = 0
else:
    prob[q][t] = max (prob[q1][t-1] * tr[q1][q] for q1 in Q) * em[q][obs[t]]
```

Style:	`if x == 0:`
Better style:	`if 0 == x:`

Coding assignment 3: linguistic insights

Tagging unseen words

- Unseen words do not have known emissions
- Use transition probabilities alone

Errors with unseen words

- Common error: nouns, verbs incorrectly tagged as prepositions, articles
- Rare error: prepositions, articles incorrectly tagged as nouns, verbs

Open-class	Noun, verb, adjective, . . .
Closed-class	Preposition, conjunction, determiner, . . .

Reducing overall error

- Tag unseen words only with open-class labels!
- How do we know which labels are open class?
  - ☛ Open-class items have a large vocabulary