Natural Language Processing

Info 159/259

Lecture 11: Neural sequence labeling (Feb 28, 2024)

Many slides & instruction ideas borrowed from:

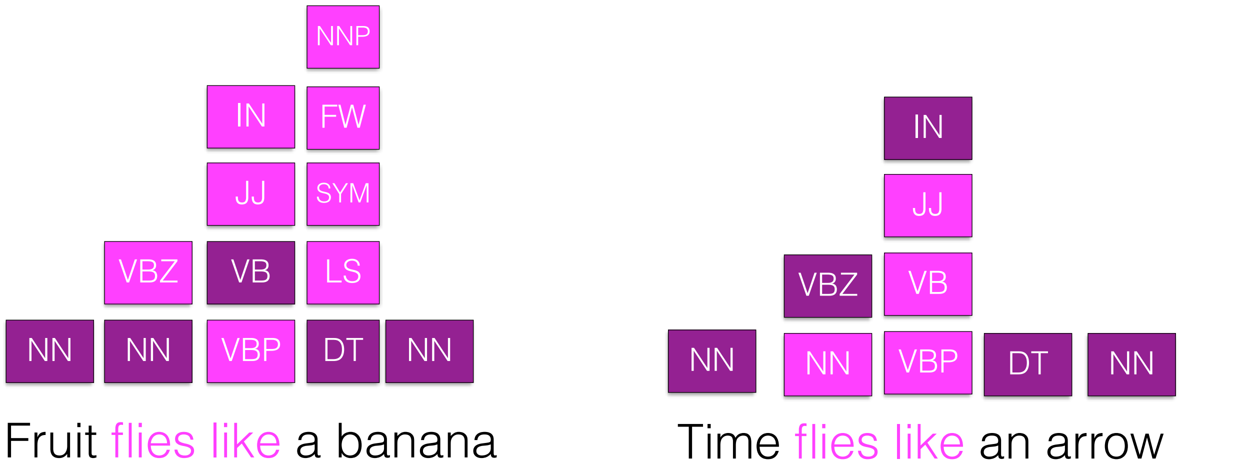
David Bamman, Dan Jurafsky & Greg Durrett

Logistics

- Exam1 grading is being finalized.
- No homework this week
    - Homework 4 will be released by Friday.
- AP1 is due this Sunday March 3
- Quiz 4 will be out this Friday afternoon (due Monday).
- Today: Reviewing HMM, followed by Neural Seq. Labeling

POS tagging

Labeling the tag that's correct for the context.



(Just tags in evidence within the Penn Treebank — more are possible!)

Sequence labeling

$$x = \{x_1, \ldots, x_n\}$$

$$y = \{y_1, \ldots, y_n\}$$

- For a set of inputs x with n sequential time steps, one corresponding label y_i for each x_i
- Model correlations in the labels y.

Generative vs. Discriminative models

- Generative models specify a joint distribution over the labels and the data. With this you could generate new data

$$P(x, y) = P(y)\, P(x \mid y)$$

- Discriminative models specify the conditional distribution of the label y given the data x. These models focus on how to discriminate between the classes

$$P(y \mid x)$$

Generative Model

$$P(x, y) = P(x \mid y)\, P(y)$$

x	time	flies	like	an	arrow
y	NN	VBZ	IN	DT	NN

$$\max_y P(x \mid y) P(y)$$

How do we parameterize these probabilities when x and y are sequences?

Estimating the seq. Prob.

$$
\begin{aligned}
P(y_i, \ldots, y_n) = &\; P(y_1) \\
&\times P(y_2 \mid y_1) \\
&\times P(y_3 \mid y_1, y_2) \\
&\cdots \\
&\times P(y_n \mid y_1, \ldots, y_{n-1})
\end{aligned}
$$

- Remember: a Markov assumption is an approximation to this exact decomposition (the chain rule of probability)

Hidden Markov Model

$$\max_y P(x \mid y) P(y)$$

Prior probability of label sequence

$$P(y) = P(y_1, \ldots, y_n)$$

$$P(y_1, \ldots, y_n) \approx \prod_{i=1}^{n} P(y_i \mid y_{i-1})$$

- We'll make a first-order Markov assumption and calculate the joint probability as the product the individual factors conditioned only on the previous tag.

Hidden Markov Model

First-order HMM

$$P(y_1,\ldots,y_n) \approx \prod_{i=1}^{n} P(y_i \mid y_{i-1})$$

Second-order HMM

$$P(y_1,\ldots,y_n) \approx \prod_{i=1}^{n} P(y_i \mid y_{i-2}, y_{i-1})$$

Third-order HMM

$$P(y_1,\ldots,y_n) \approx \prod_{i=1}^{n} P(y_i \mid y_{i-3}, y_{i-2}, y_{i-1})$$

Hidden Markov Model

$$\boxed{\max_y P(x \mid y)P(y)}$$

$$P(x \mid y) = P(x_1, \ldots, x_n \mid y_1, \ldots, y_n)$$

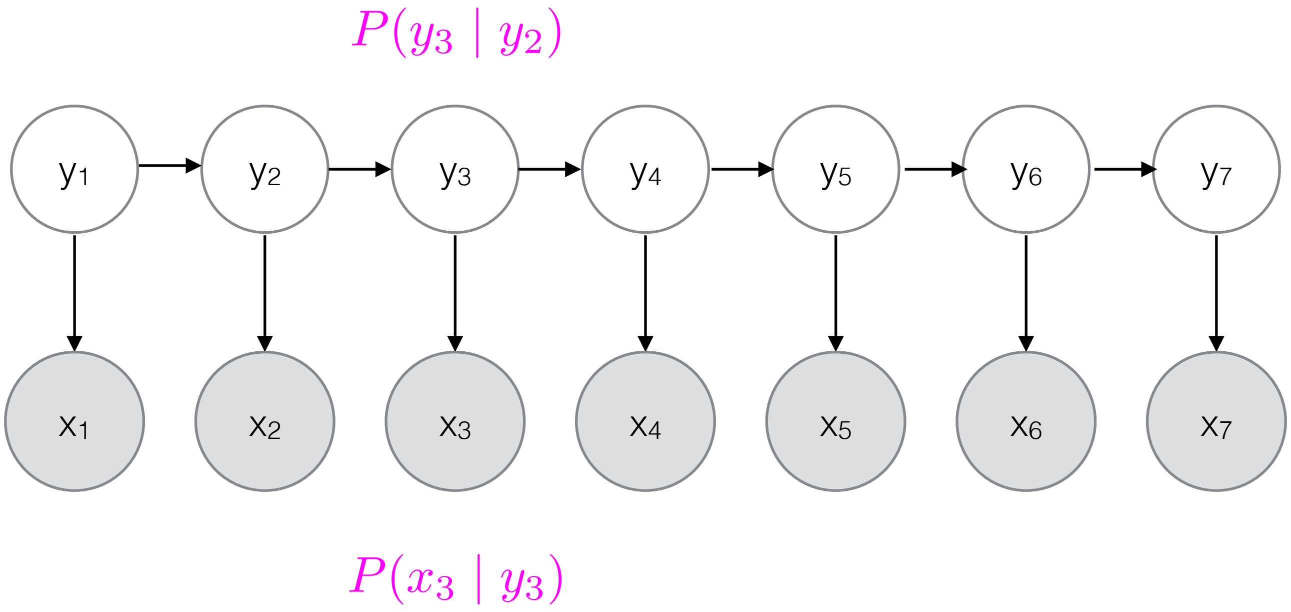
$$P(x_1, \ldots, x_n \mid y_1, \ldots, y_n) \approx \prod_{i=1}^{N} P(x_i \mid y_i)$$

- Here again we'll make a strong assumption: the probability of the word we see at a given time step is only dependent on its own label, no matter of the Markov order used for P(y).

HMM

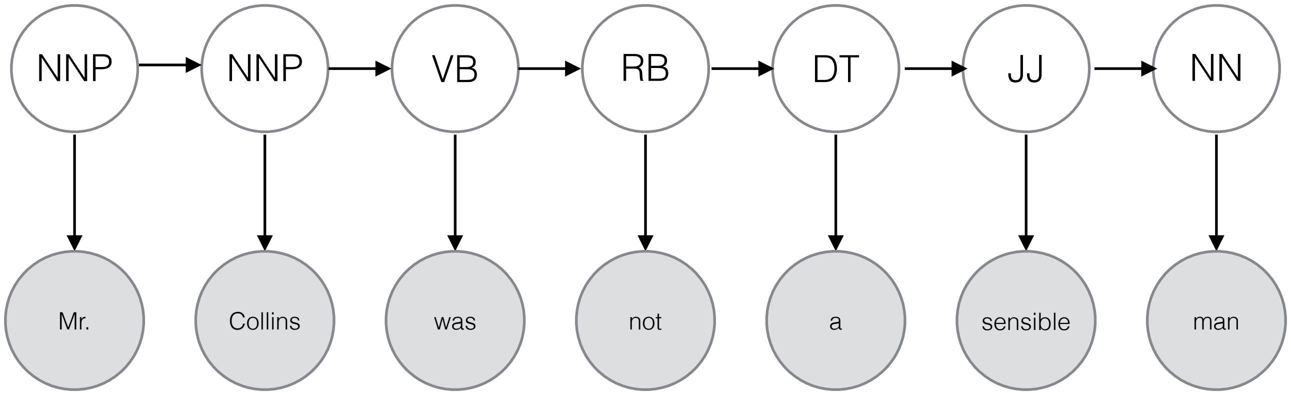
$$P(x_1,..,x_n,y_1,..,y_n) \approx \prod_{i=1}^{n} P(y_i \mid y_{i-1}) \prod_{i=1}^{n} P(x_i \mid y_i)$$

HMM



HMM

$P(VB \mid NNP)$



$P(was \mid VB)$

Parameter estimation

$$P(y_t \mid y_{t-1}) \qquad \frac{c(y_1, y_2)}{c(y_1)}$$

MLE for both is just counting
and normalizing

$$P(x_t \mid y_t) \qquad \frac{c(x, y)}{c(y)}$$

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.

Decoding

- **Decoding:** Finding the optimal path for a sequence using transition and emission parameters.

- **Greedy decoding**: proceed left to right, committing to the best tag for each time step (given the sequence seen so far)

Fruit	flies	like	a	banana
NN	VB	IN	DT	NN

Decoding

DT	NN	VBD	IN	DT	NN	???
The	horse	raced	past	the	barn	fell

Decoding

DT	NN	VBD	IN	DT	NN	???
The	horse	raced	past	the	barn	fell
DT	NN	VBN	IN	DT	NN	VBD

Information later on in the sentence can influence the best tags earlier on.

All paths

END							
DT							
NNP							
VB							
NN							
MD							
START							
	^	Janet	will	back	the	bill	$

Ideally, what we want is to calculate the joint probability of each path and pick the one with the highest probability. But for N time steps and K labels, number of possible paths = K^N

5 word sentence with 45 Penn Treebank tags

45^5 = 184,528,125 different paths

45^{20} = 1.16e33 different paths

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 discard all non-optimal paths up to label L at time T

END							
DT							
NNP							
VB							
NN							
MD							
START							
	^	Janet	will	back	the	bill	$

- At each time step t ending in label K, we find the max probability of any path that led to that state

END		
DT		$v_1(DT)$
NNP		$v_1(NNP)$
VB		$v_1(VB)$
NN		$v_1(NN)$
MD		$v_1(MD)$
START		
		Janet

What's the HMM probability of ending in Janet = NNP?

$$P(y_t \mid y_{t-1})P(x_t \mid y_t)$$

$$P(\text{NNP} \mid \text{START})P(Janet \mid \text{NNP})$$

END		
DT		v_1(DT)
NNP		v_1(NNP)
VB		v_1(VB)
NN		v_1(NN)
MD		v_1(MD)
START		
		Janet

Best path through time step 1 ending in tag y (trivially - best path for all is just START)

$$v_1(y) = \max_{u \in \mathcal{Y}} \left[P(y_t = y \mid y_{t-1} = u) P(x_t \mid y_t = y) \right]$$

END			
DT		$v_1(DT)$	$v_2(DT)$
NNP		$v_1(NNP)$	$v_2(NNP)$
VB		$v_1(VB)$	$v_2(VB)$
NN		$v_1(NN)$	$v_2(NN)$
MD		$v_1(MD)$	$v_2(MD)$
START			
		Janet	will

What's the max HMM probability of ending in will = MD?

First, what's the HMM probability of a single path ending in will = MD?

END			
DT		v_1(DT)	v_2(DT)
NNP		v_1(NNP)	v_2(NNP)
VB		v_1(VB)	v_2(VB)
NN		v_1(NN)	v_2(NN)
MD		v_1(MD)	v_2(MD)
START			
		Janet	will

$$P(y_1 \mid START)P(x_1 \mid y_1) \times P(y_2 = \mathrm{MD} \mid y_1)P(x_2 \mid y_2 = \mathrm{MD})$$

END			
DT		v_1(DT)	v_2(DT)
NNP		v_1(NNP)	v_2(NNP)
VB		v_1(VB)	v_2(VB)
NN		v_1(NN)	v_2(NN)
MD		v_1(MD)	v_2(MD)
START			
		Janet	will

Best path through time step 2 ending in tag MD

$$P(\mathrm{DT} \mid \mathrm{START}) \times P(\mathit{Janet} \mid \mathrm{DT}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{DT}) \times P(\mathit{will} \mid y_t = \mathrm{MD})$$

$$P(\mathrm{NNP} \mid \mathrm{START}) \times P(\mathit{Janet} \mid \mathrm{NNP}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{NNP}) \times P(\mathit{will} \mid y_t = \mathrm{MD})$$

$$P(\mathrm{VB} \mid \mathrm{START}) \times P(\mathit{Janet} \mid \mathrm{VB}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{VB}) \times P(\mathit{will} \mid y_t = \mathrm{MD})$$

$$P(\mathrm{NN} \mid \mathrm{START}) \times P(\mathit{Janet} \mid \mathrm{NN}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{NN}) \times P(\mathit{will} \mid y_t = \mathrm{MD})$$

$$P(\mathrm{MD} \mid \mathrm{START}) \times P(\mathit{Janet} \mid \mathrm{MD}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{MD}) \times P(\mathit{will} \mid y_t = \mathrm{MD})$$

Let's say the best path ending y_2=MD includes y_1=NNP, with probability 0.0090.

Under our first-order Markov assumption, *if* y_2=MD is in the best path for the complete sequence, y_1=NNP must be as well. That means we can forget every other path ending in y_2=MD that does not have y_1=NNP.

END			
DT		v_1(DT)	v_2(DT)
NNP		v_1(NNP)	v_2(NNP)
VB		v_1(VB)	v_2(VB)
NN		v_1(NN)	v_2(NN)
MD		v_1(MD)	v_2(MD)
START			
		Janet	will

0.0003	$P(\mathrm{DT} \mid \mathrm{START}) \times P(Janet \mid \mathrm{DT}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{DT}) \times P(will \mid y_t = \mathrm{MD})$
0.0090	$P(\mathrm{NNP} \mid \mathrm{START}) \times P(Janet \mid \mathrm{NNP}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{NNP}) \times P(will \mid y_t = \mathrm{MD})$
0.0001	$P(\mathrm{VB} \mid \mathrm{START}) \times P(Janet \mid \mathrm{VB}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{VB}) \times P(will \mid y_t = \mathrm{MD})$
0.0045	$P(\mathrm{NN} \mid \mathrm{START}) \times P(Janet \mid \mathrm{NN}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{NN}) \times P(will \mid y_t = \mathrm{MD})$
0.0002	$P(\mathrm{MD} \mid \mathrm{START}) \times P(Janet \mid \mathrm{MD}) \times P(y_t = \mathrm{MD} \mid P(y_{t-1} = \mathrm{MD}) \times P(will \mid y_t = \mathrm{MD})$

None of the grey out paths could *possibly* be in the final optimal path, so we can forget them going forward.

To calculate this full probability, notice that we can reuse information we've already computed.

$$\underbrace{P(\text{DT} \mid \text{START}) \times P(Janet \mid \text{DT})}_{v_1(DT)} \times P(y_t = MD \mid P(y_{t-1} = DT) \times P(will \mid y_t = MD)$$

$$\underbrace{P(\text{NNP} \mid \text{START}) \times P(Janet \mid \text{NNP})}_{v_1(NNP)} \times P(y_t = MD \mid P(y_{t-1} = NNP) \times P(will \mid y_t = MD)$$

$$\underbrace{P(\text{VB} \mid \text{START}) \times P(Janet \mid \text{VB})}_{v_1(VB)} \times P(y_t = MD \mid P(y_{t-1} = VB) \times P(will \mid y_t = MD)$$

$\ldots$

END			
DT		v_1(DT)	v_2(DT)
NNP		v_1(NNP)	v_2(NNP)
VB		v_1(VB)	v_2(VB)
NN		v_1(NN)	v_2(NN)
MD		v_1(MD)	v_2(MD)
START			
		Janet	will

$$v_t(y) = \max_{u \in \mathcal{Y}} \left[v_{t-1}(u) \times P(y_t = y \mid y_{t-1} = u) P(x_t \mid y_t = y) \right]$$

END			
DT		v_1(DT)	v_2(DT)
NNP		v_1(NNP)	v_2(NNP)
VB		v_1(VB)	v_2(VB)
NN		v_1(NN)	v_2(NN)
MD		v_1(MD)	v_2(MD)
START			
		Janet	will

Every y at time step t may have a different u at time step t-1 that leads to its max.

Once we've determined that u for each y, we can forget all of the other values of u for that each y, since we know they cannot be on the optimal path for the entire sequence.

$$v_t(y) = \max_{u \in \mathcal{Y}} \left[v_{t-1}(u) \times P(y_t = y \mid y_{t-1} = u) P(x_t \mid y_t = y) \right]$$

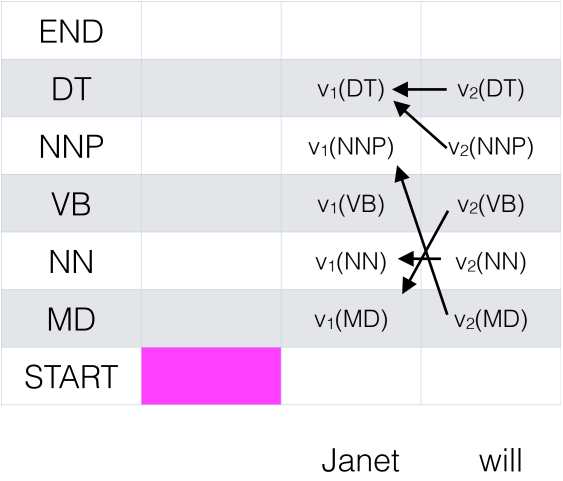
END				
DT		$v_1(DT)$	$v_2(DT)$	$v_3(DT)$
NNP		$v_1(NNP)$	$v_2(NNP)$	$v_3(NNP)$
VB		$v_1(VB)$	$v_2(VB)$	$v_3(VB)$
NN		$v_1(NN)$	$v_2(NN)$	$v_3(NN)$
MD		$v_1(MD)$	$v_2(MD)$	$v_3(MD)$
START				
		Janet	will	back

25 paths ending in back = VB

$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = DT)P(x_2 = will \mid y_2 = DT)P(y_2 = DT \mid y_1 = DT)P(x_1 = Janet \mid y_1 = DT)P(y_1 = DT \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = DT)P(x_2 = will \mid y_2 = DT)P(y_2 = DT \mid y_1 = NNP)P(x_1 = Janet \mid y_1 = NNP)P(y_1 = NNP \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = DT)P(x_2 = will \mid y_2 = DT)P(y_2 = DT \mid y_1 = VB)P(x_1 = Janet \mid y_1 = VB)P(y_1 = VB \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = DT)P(x_2 = will \mid y_2 = DT)P(y_2 = DT \mid y_1 = NN)P(x_1 = Janet \mid y_1 = NN)P(y_1 = NN \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = DT)P(x_2 = will \mid y_2 = DT)P(y_2 = DT \mid y_1 = MD)P(x_1 = Janet \mid y_1 = MD)P(y_1 = MD \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NNP)P(x_2 = will \mid y_2 = NNP)P(y_2 = NNP \mid y_1 = DT)P(x_1 = Janet \mid y_1 = DT)P(y_1 = DT \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NNP)P(x_2 = will \mid y_2 = NNP)P(y_2 = NNP \mid y_1 = NNP)P(x_1 = Janet \mid y_1 = NNP)P(y_1 = NNP \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NNP)P(x_2 = will \mid y_2 = NNP)P(y_2 = NNP \mid y_1 = VB)P(x_1 = Janet \mid y_1 = VB)P(y_1 = VB \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NNP)P(x_2 = will \mid y_2 = NNP)P(y_2 = NNP \mid y_1 = NN)P(x_1 = Janet \mid y_1 = NN)P(y_1 = NN \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NNP)P(x_2 = will \mid y_2 = NNP)P(y_2 = NNP \mid y_1 = MD)P(x_1 = Janet \mid y_1 = MD)P(y_1 = MD \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = VB)P(x_2 = will \mid y_2 = VB)P(y_2 = VB \mid y_1 = DT)P(x_1 = Janet \mid y_1 = DT)P(y_1 = DT \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = VB)P(x_2 = will \mid y_2 = VB)P(y_2 = VB \mid y_1 = NNP)P(x_1 = Janet \mid y_1 = NNP)P(y_1 = NNP \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = VB)P(x_2 = will \mid y_2 = VB)P(y_2 = VB \mid y_1 = VB)P(x_1 = Janet \mid y_1 = VB)P(y_1 = VB \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = VB)P(x_2 = will \mid y_2 = VB)P(y_2 = VB \mid y_1 = NN)P(x_1 = Janet \mid y_1 = NN)P(y_1 = NN \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = VB)P(x_2 = will \mid y_2 = VB)P(y_2 = VB \mid y_1 = MD)P(x_1 = Janet \mid y_1 = MD)P(y_1 = MD \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NN)P(x_2 = will \mid y_2 = NN)P(y_2 = NN \mid y_1 = DT)P(x_1 = Janet \mid y_1 = DT)P(y_1 = DT \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NN)P(x_2 = will \mid y_2 = NN)P(y_2 = NN \mid y_1 = NNP)P(x_1 = Janet \mid y_1 = NNP)P(y_1 = NNP \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NN)P(x_2 = will \mid y_2 = NN)P(y_2 = NN \mid y_1 = VB)P(x_1 = Janet \mid y_1 = VB)P(y_1 = VB \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NN)P(x_2 = will \mid y_2 = NN)P(y_2 = NN \mid y_1 = NN)P(x_1 = Janet \mid y_1 = NN)P(y_1 = NN \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NN)P(x_2 = will \mid y_2 = NN)P(y_2 = NN \mid y_1 = MD)P(x_1 = Janet \mid y_1 = MD)P(y_1 = MD \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = MD)P(x_2 = will \mid y_2 = MD)P(y_2 = MD \mid y_1 = DT)P(x_1 = Janet \mid y_1 = DT)P(y_1 = DT \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = MD)P(x_2 = will \mid y_2 = MD)P(y_2 = MD \mid y_1 = NNP)P(x_1 = Janet \mid y_1 = NNP)P(y_1 = NNP \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = MD)P(x_2 = will \mid y_2 = MD)P(y_2 = MD \mid y_1 = VB)P(x_1 = Janet \mid y_1 = VB)P(y_1 = VB \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = MD)P(x_2 = will \mid y_2 = MD)P(y_2 = MD \mid y_1 = NN)P(x_1 = Janet \mid y_1 = NN)P(y_1 = NN \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = MD)P(x_2 = will \mid y_2 = MD)P(y_2 = MD \mid y_1 = MD)P(x_1 = Janet \mid y_1 = MD)P(y_1 = MD \mid START)$$

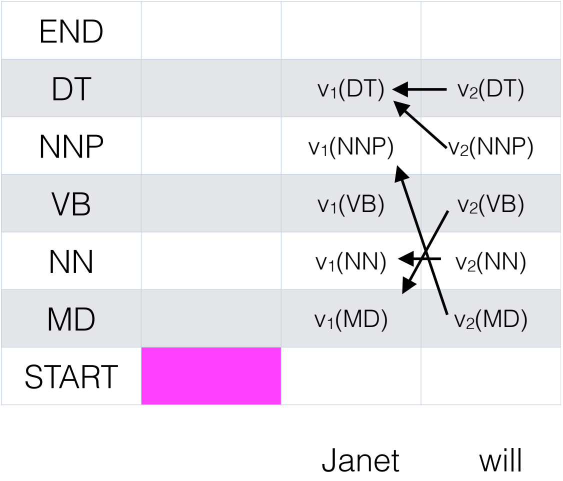
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = DT)P(x_2 = will \mid y_2 = DT)P(y_2 = DT \mid y_1 = DT)P(x_1 = Janet \mid y_1 = DT)P(y_1 = DT \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NNP)P(x_2 = will \mid y_2 = NNP)P(y_2 = NNP \mid y_1 = DT)P(x_1 = Janet \mid y_1 = DT)P(y_1 = DT \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = MD)P(x_2 = will \mid y_2 = MD)P(y_2 = MD \mid y_1 = NNP)P(x_1 = Janet \mid y_1 = NNP)P(y_1 = NNP \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NN)P(x_2 = will \mid y_2 = NN)P(y_2 = NN \mid y_1 = NN)P(x_1 = Janet \mid y_1 = NN)P(y_1 = NN \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = VB)P(x_2 = will \mid y_2 = VB)P(y_2 = VB \mid y_1 = MD)P(x_1 = Janet \mid y_1 = MD)P(y_1 = MD \mid START)$$

In calculating the best path ending in x_3=back and y_3=VB, we can forget every other path that we've already determined to be suboptimal.



$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = DT)P(x_2 = will \mid y_2 = DT)P(y_2 = DT \mid y_1 = DT)P(x_1 = Janet \mid y_1 = DT)P(y_1 = DT \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NNP)P(x_2 = will \mid y_2 = NNP)P(y_2 = NNP \mid y_1 = DT)P(x_1 = Janet \mid y_1 = DT)P(y_1 = DT \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = MD)P(x_2 = will \mid y_2 = MD)P(y_2 = MD \mid y_1 = NNP)P(x_1 = Janet \mid y_1 = NNP)P(y_1 = NNP \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = NN)P(x_2 = will \mid y_2 = NN)P(y_2 = NN \mid y_1 = NN)P(x_1 = Janet \mid y_1 = NN)P(y_1 = NN \mid START)$$
$$P(x_3 = back \mid y_3 = VB)P(y_3 = VB \mid y_2 = VB)P(x_2 = will \mid y_2 = VB)P(y_2 = VB \mid y_1 = MD)P(x_1 = Janet \mid y_1 = MD)P(y_1 = MD \mid START)$$

In calculating the best path ending in x_3=back and y_3=VB, we can forget every other path that we've already determined to be suboptimal.



END				
DT		$v_1(DT)$	$v_2(DT)$	$v_3(DT)$
NNP		$v_1(NNP)$	$v_2(NNP)$	$v_3(NNP)$
VB		$v_1(VB)$	$v_2(VB)$	$v_3(VB)$
NN		$v_1(NN)$	$v_2(NN)$	$v_3(NN)$
MD		$v_1(MD)$	$v_2(MD)$	$v_3(MD)$
START				

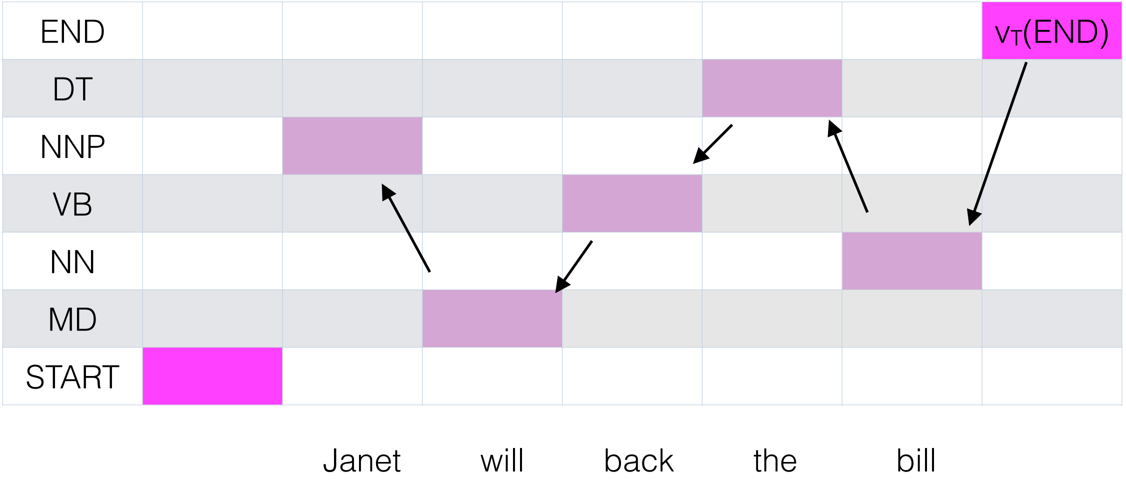
Janet will back

So for every label at every time step, we only need to keep track of which label at the previous time step *t*-1 led to the highest joint probability at that time step *t*.

END						
DT		$v_1(DT)$	$v_2(DT)$	$v_3(DT)$	$v_4(DT)$	$v_5(DT)$
NNP		$v_1(NNP)$	$v_2(NNP)$	$v_3(NNP)$	$v_4(NNP)$	$v_5(NNP)$
VB		$v_1(VB)$	$v_2(VB)$	$v_3(VB)$	$v_4(MD)$	$v_5(MD)$
NN		$v_1(NN)$	$v_2(NN)$	$v_3(NN)$	$v_4(NN)$	$v_5(NN)$
MD		$v_1(MD)$	$v_2(MD)$	$v_3(MD)$	$v_4(MD)$	$v_5(MD)$
START						
		Janet	will	back	the	bill

END							$v_T(END)$
DT		$v_1(DT)$	$v_2(DT)$	$v_3(DT)$	$v_4(DT)$	$v_5(DT)$	
NNP		$v_1(NNP)$	$v_2(NNP)$	$v_3(NNP)$	$v_4(NNP)$	$v_5(NNP)$	
VB		$v_1(VB)$	$v_2(VB)$	$v_3(VB)$	$v_4(MD)$	$v_5(MD)$	
NN		$v_1(NN)$	$v_2(NN)$	$v_3(NN)$	$v_4(NN)$	$v_5(NN)$	
MD		$v_1(MD)$	$v_2(MD)$	$v_3(MD)$	$v_4(MD)$	$v_5(MD)$	
START							
		Janet	will	back	the	bill	

$v_T(END)$ encodes the best path through the entire sequence



For each timestep t + label, keep track of the max element from t-1 to reconstruct best path

Named entity recognition

[tim cook]$_{\text{PER}}$ is the ceo of [apple]$_{\text{ORG}}$

- Identifying spans of text that correspond to typed entities
- Another application of sequence labeling

Named entity recognition

Type	Tag	Sample Categories	Example sentences
People	PER	people, characters	**Turing** is a giant of computer science.
Organization	ORG	companies, sports teams	The **IPCC** warned about the cyclone.
Location	LOC	regions, mountains, seas	The **Mt. Sanitas** loop is in **Sunshine Canyon**.
Geo-Political Entity	GPE	countries, states, provinces	**Palo Alto** is raising the fees for parking.
Facility	FAC	bridges, buildings, airports	Consider the **Golden Gate Bridge**.
Vehicles	VEH	planes, trains, automobiles	It was a classic **Ford Falcon**.

Figure 17.1 A list of generic named entity types with the kinds of entities they refer to.

ACE NER categories (+weapon)

Named entity recognition

- GENIA corpus of MEDLINE abstracts (biomedical)

We have shown that [interleukin-1]PROTEIN ([IL-1]PROTEIN) and [IL-2]PROTEIN control [IL-2 receptor alpha (IL-2R alpha) gene]DNA transcription in [CD4-CD8- murine T lymphocyte precursors]CELL LINE

protein
cell line
cell type
DNA
RNA

http://www.aclweb.org/anthology/W04-1213

BIO notation

B-PERS	I-PERS	O	O	O	O	B-ORG
tim	cook	is	the	ceo	of	apple

tim cook is the ceo of apple

- **B**eginning of entity
- **I**nside entity
- **O**utside entity

[tim cook]$_{\text{PER}}$ is the ceo of [apple]$_{\text{ORG}}$

Named entity recognition

B-PER B-PER

After he saw Harry Tom went to the store

Evaluation

- We evaluate NER with precision/recall/F1 over typed chunks.

Evaluation

	1	2	3	4	5	6	7
	tim	cook	is	the	CEO	of	Apple
gold	B-PER	I-PER	O	O	O	O	B-ORG
system	B-PER	O	O	O	B-PER	O	B-ORG

<start, end, type>

Precision	1/3
Recall	1/2

gold

<1,2,PER>
<7,7,ORG>

system

<1,1,PER>
<5,5,PER>
<7,7,ORG>

Sequence labeling

$$x = \{x_1, \ldots, x_n\}$$

$$y = \{y_1, \ldots, y_n\}$$

- For a set of inputs x with n sequential time steps, one corresponding label y_i for each x_i
- Model correlations in the labels y.

Generative vs. Discriminative models

- Generative models specify a joint distribution over the labels and the data. With this you could generate new data.

$$P(x, y) = P(y)\, P(x \mid y)$$

- Discriminative models specify the conditional distribution of the label y given the data x. These models focus on how to discriminate between the classes

$$P(y \mid x)$$

Maximum Entropy Markov Model (MEMM)

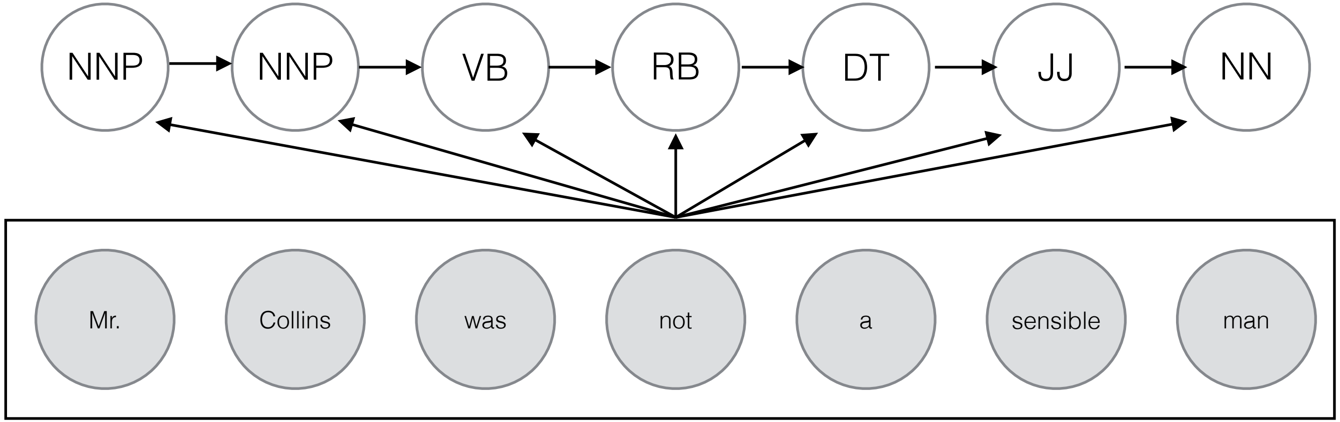
General maxent form

$$\arg\max_y P(y \mid x, \beta)$$

Maxent with first-order Markov assumption: Maximum Entropy Markov Model

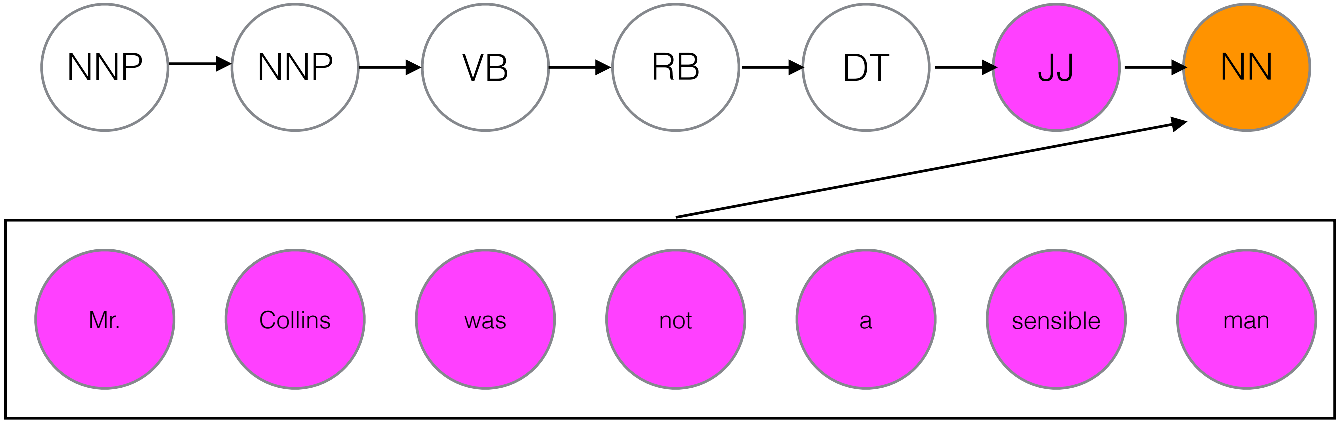
$$\arg\max_y \prod_{i=1}^{n} P(y_i \mid y_{i-1}, x)$$

MEMM



MEMMs condition on the *entire* input

MEMM



Features

$$f(y_i, y_{i-1}; x_1, \ldots, x_n)$$

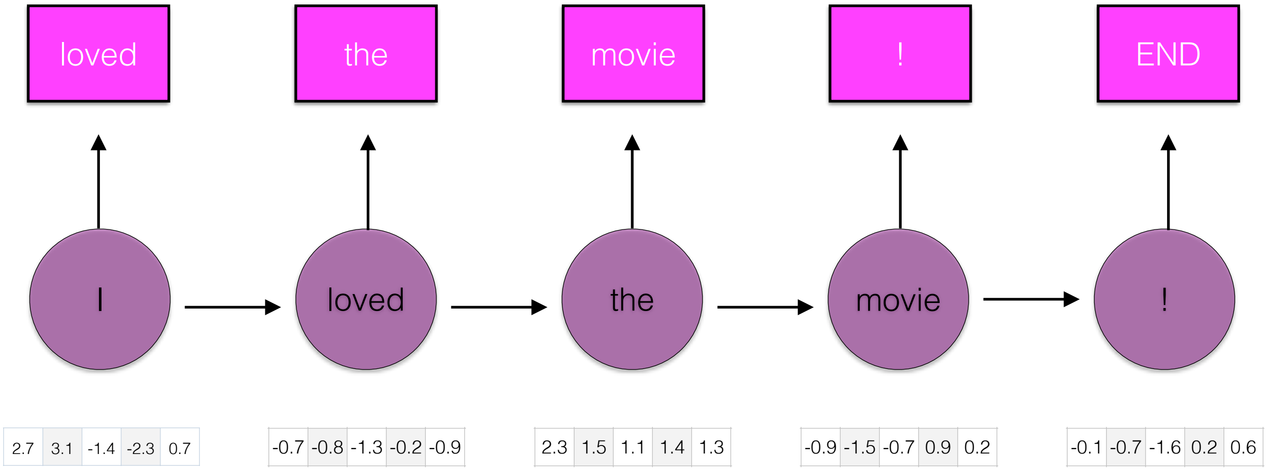
Features are scoped over the previous predicted tag and the entire observed input

feature	example
x_i = man	1
y_{i-1} = JJ	1
i=n (last word of sentence)	1
x_i ends in -ly	0

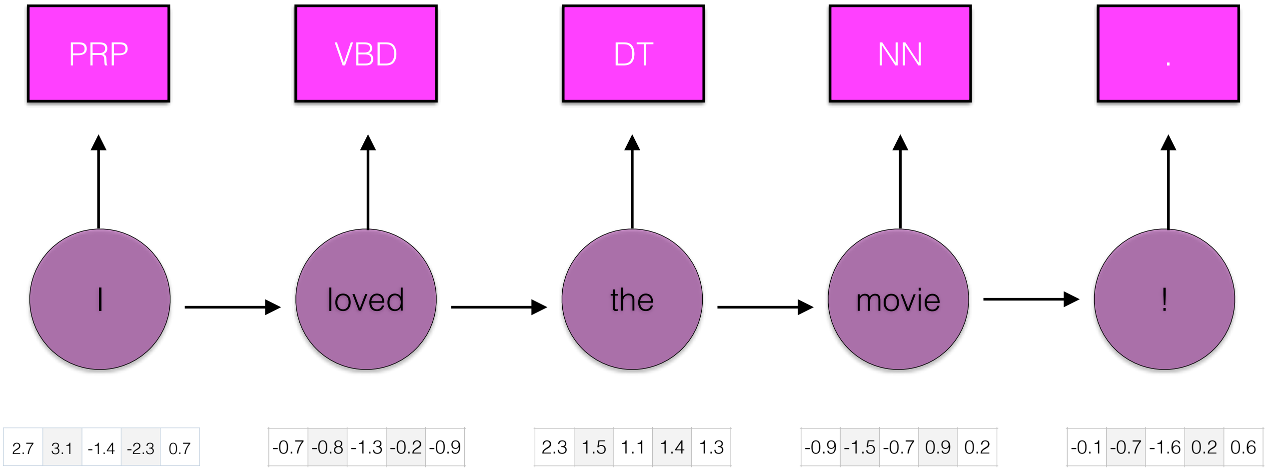
Recurrent neural network

- RNNs allow arbitarily-sized conditioning contexts and condition on the entire sequence history.

RNNs for language modeling are already performing a kind of sequence labeling: at each time step, predict the word from $\mathcal{V}$ conditioned on the context



For POS tagging, predict the tag from $\boldsymbol{y}$ conditioned on the context



RNNs for POS

NN TO VB

will to fight

- To make a prediction for y_t, RNNs condition on all input seen through time t (x_1, …, x_t)
- But knowing something about the future can help (x_{t+1}, …, x_n)

Bidirectional RNN

- A powerful alternative is make predictions conditioning both on the past and the future.

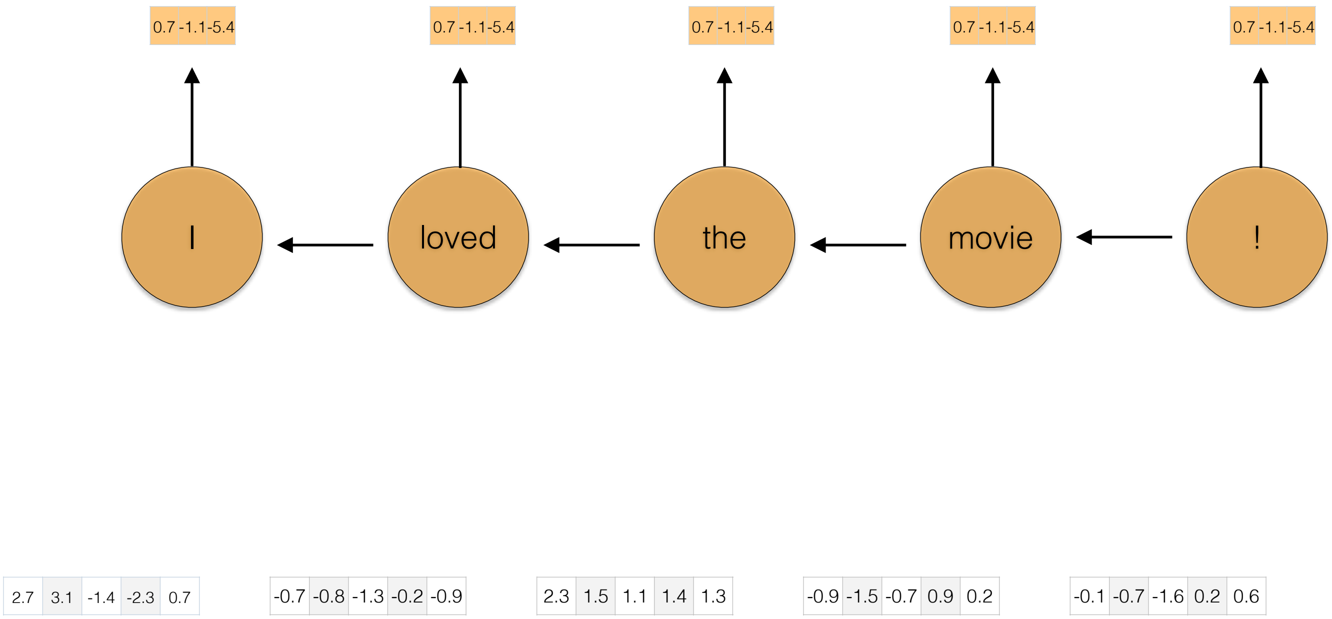
- Two RNNs

  - One running left-to-right
  - One right-to-left

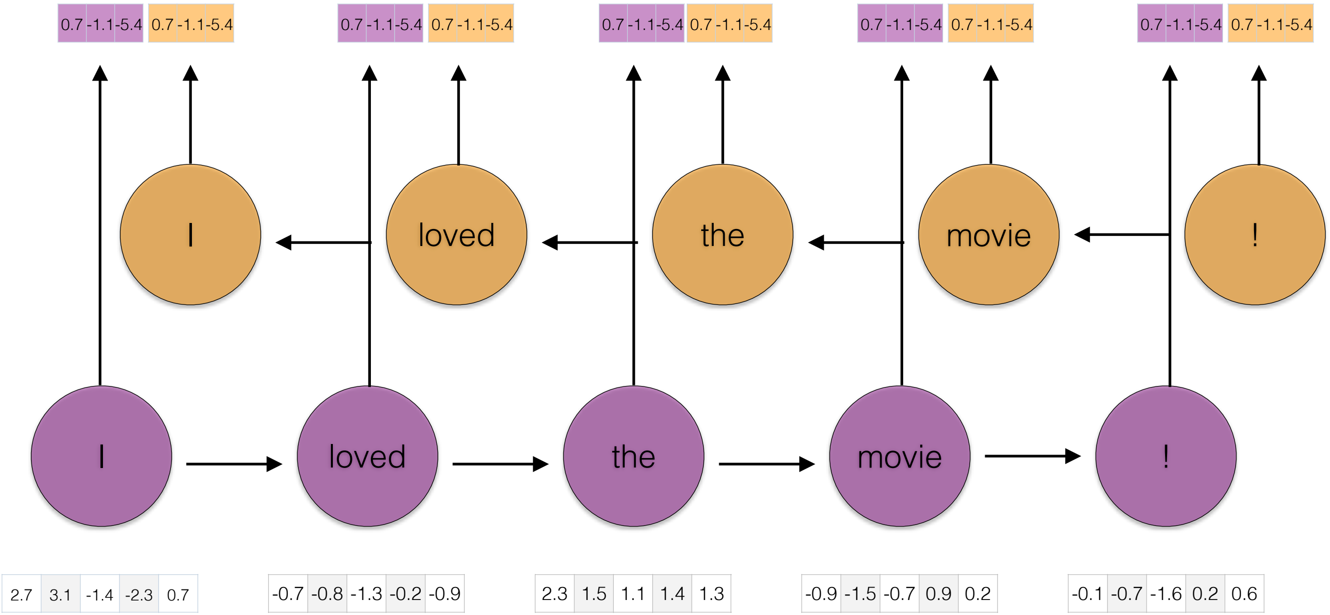
- Each produces an output vector at each time step, which we concatenate

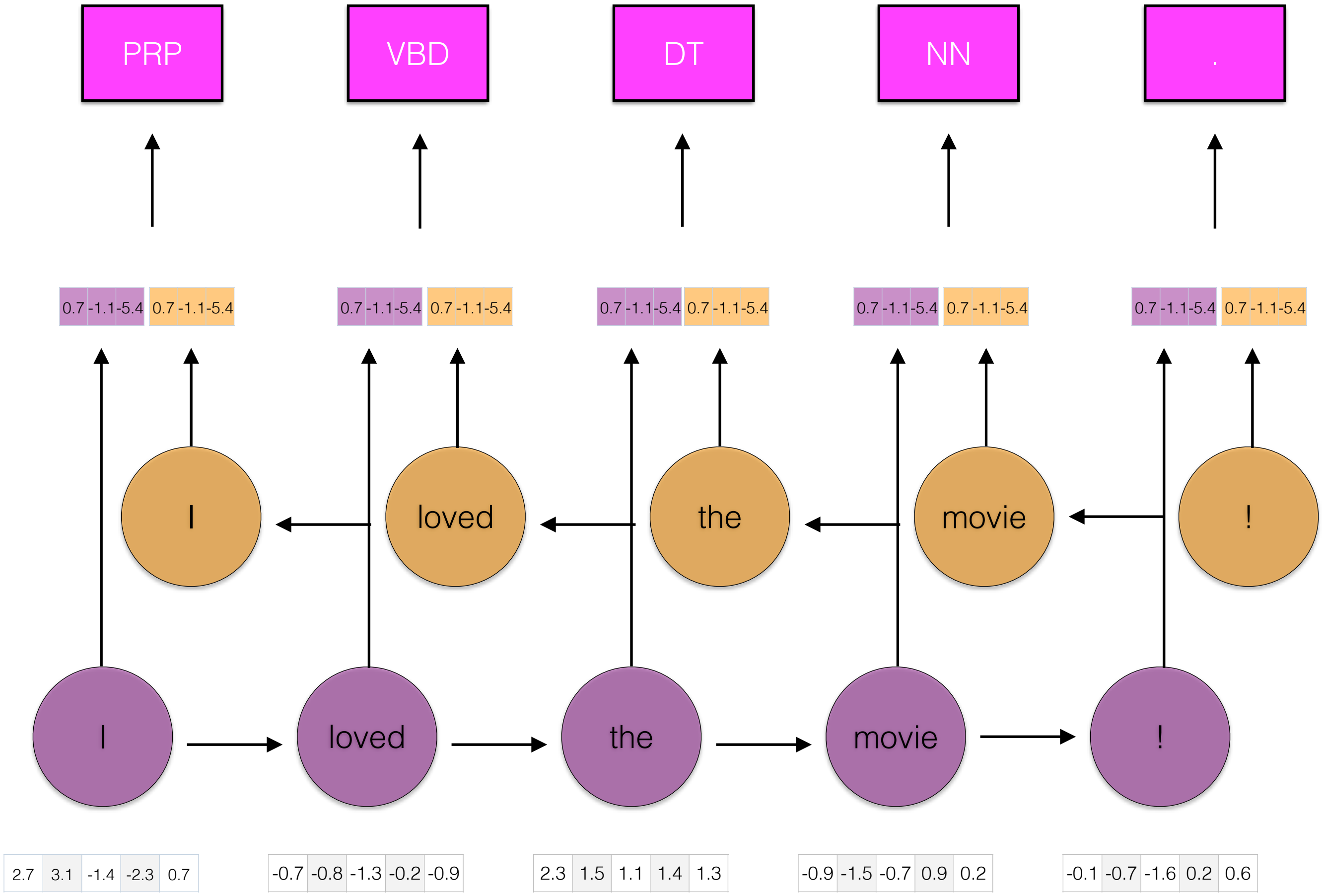
Bidirectional RNN

backward RNN



Bidirectional RNN

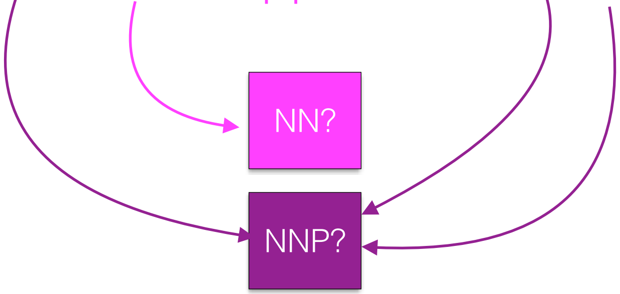



PRP
VBD
DT
NN
.
0.7 -1.1 -5.4 0.7 -1.1 -5.4
0.7 -1.1 -5.4 0.7 -1.1 -5.4
0.7 -1.1 -5.4 0.7 -1.1 -5.4
0.7 -1.1 -5.4 0.7 -1.1 -5.4
0.7 -1.1 -5.4 0.7 -1.1 -5.4
I
loved
the
movie
!
I
loved
the
movie
!
2.7 3.1 -1.4 -2.3 0.7
-0.7 -0.8 -1.3 -0.2 -0.9
2.3 1.5 1.1 1.4 1.3
-0.9 -1.5 -0.7 0.9 0.2
-0.1 -0.7 -1.6 0.2 0.6

RNNs for POS tagging

amazon and spotify's streaming services are going to

devour apple and its music purchasing model



RNNs for POS tagging

amazon and spotify's streaming services are going to

devour apple and its music purchasing model

Prediction: Can the information from far away get to the time step that needs it?

Training: Can error reach that far back during backpropagation?

RNNs

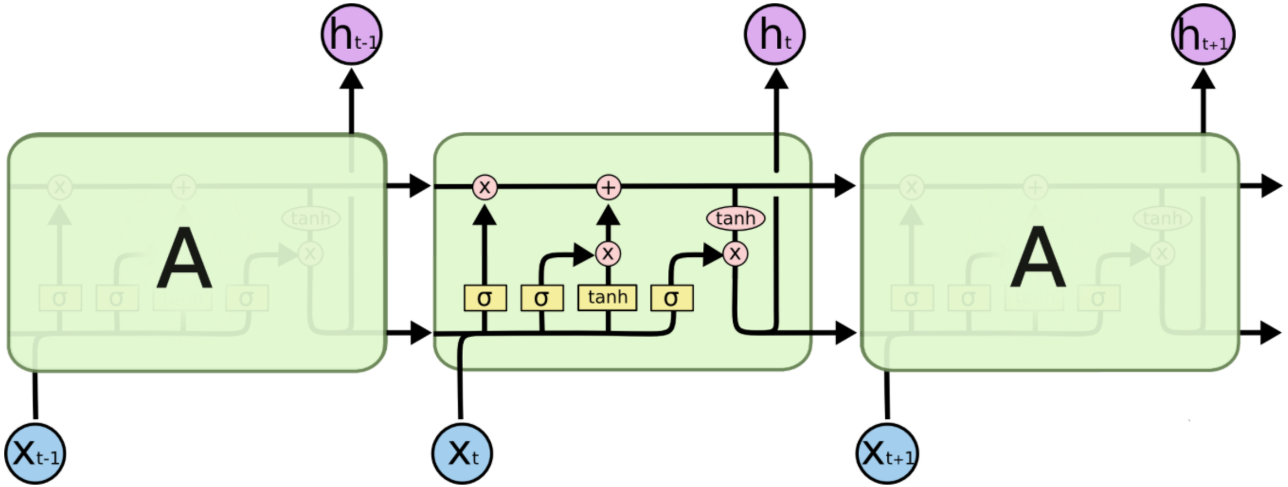
- Recurrent networks are deep in that they involve one “layer” for each time step (e.g., words in a sentence)

- Vanishing gradient problem: as error is back propagated through the layers of a deep network, they tend toward 0.

Long short-term memory network (LSTM)

- Designed to account for the vanishing gradient problem
- Basic idea: split the *s* vector propagated between time steps into a memory component and a hidden state component

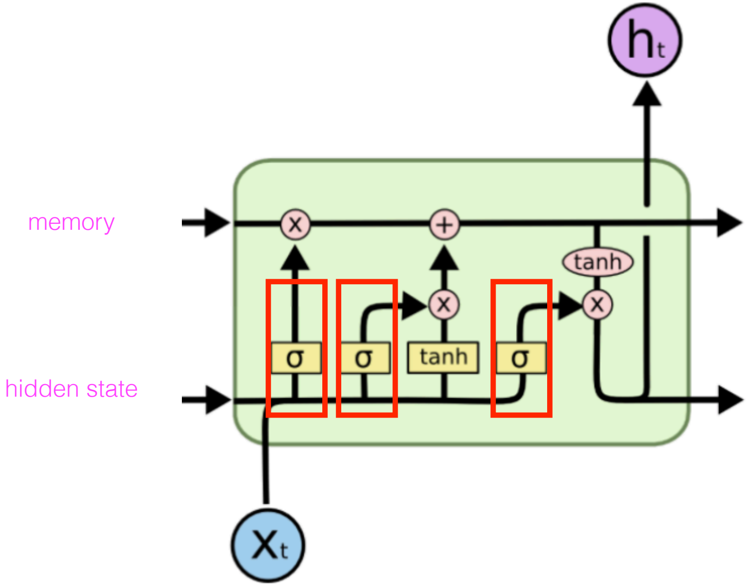
LSTMs



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gates

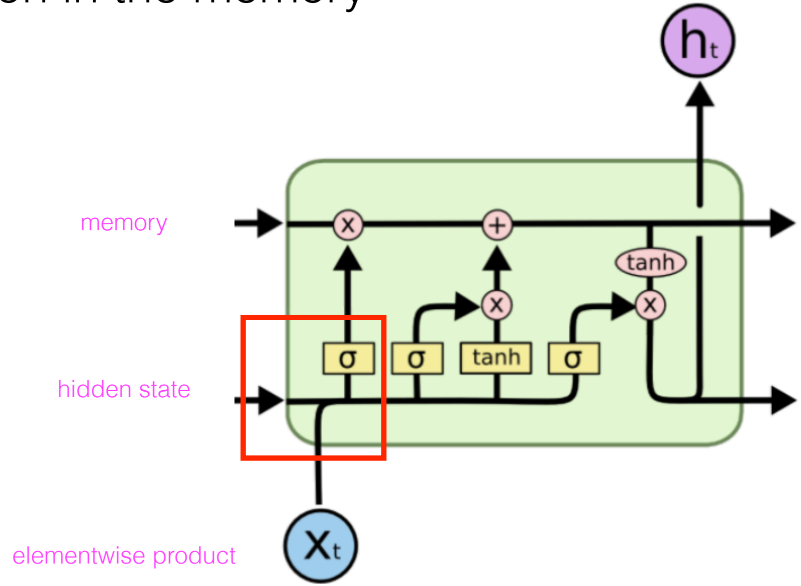
- LSTMs gates control the flow of information



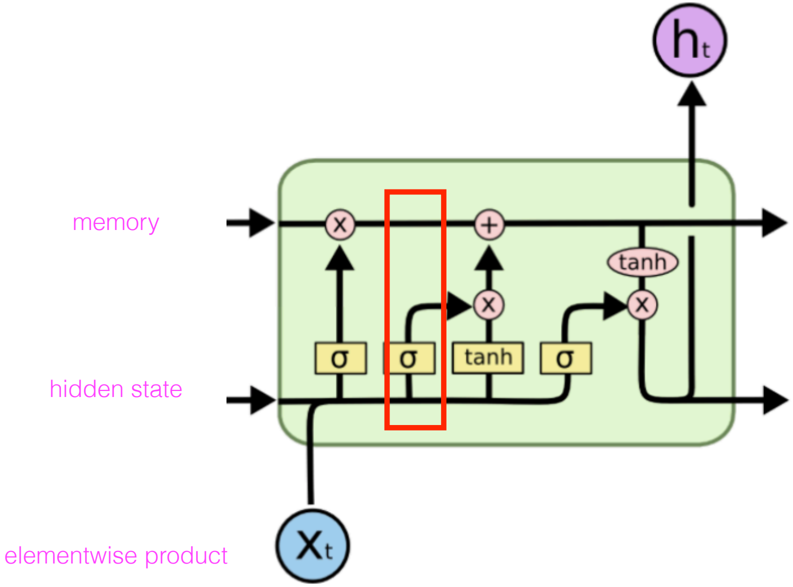
- A sigmoid squashes its input to between 0 and 1
- By multiplying the output of a sigmoid element-wise with another vector, we forget information in the vector (if multiplied by 0) or allow it to pass (if multiplied by 1)

input	3.7	1.4	-0.7	-1.4	7.8
gate	0.01	0.99	0.5	0.98	0.01
output	0.03	1.4	-0.35	-1.38	0.08

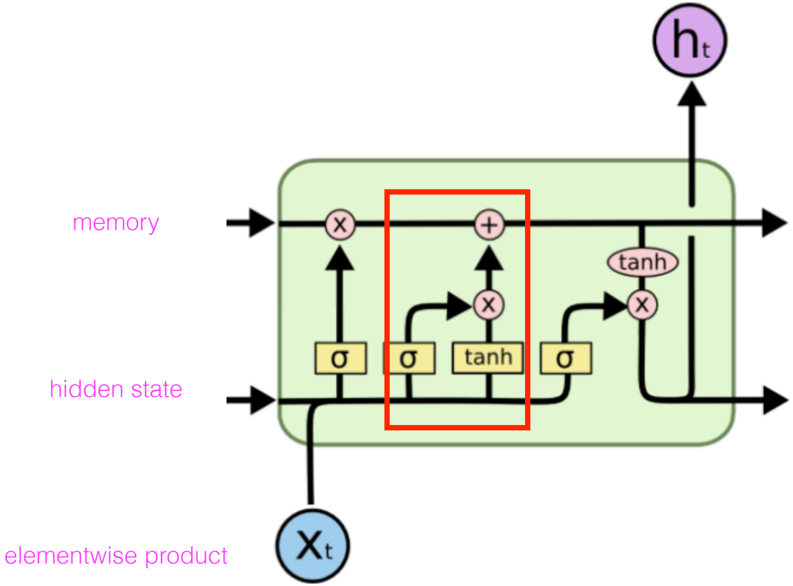
Forget gate: as a function of the previous hidden state and current input, forget information in the memory



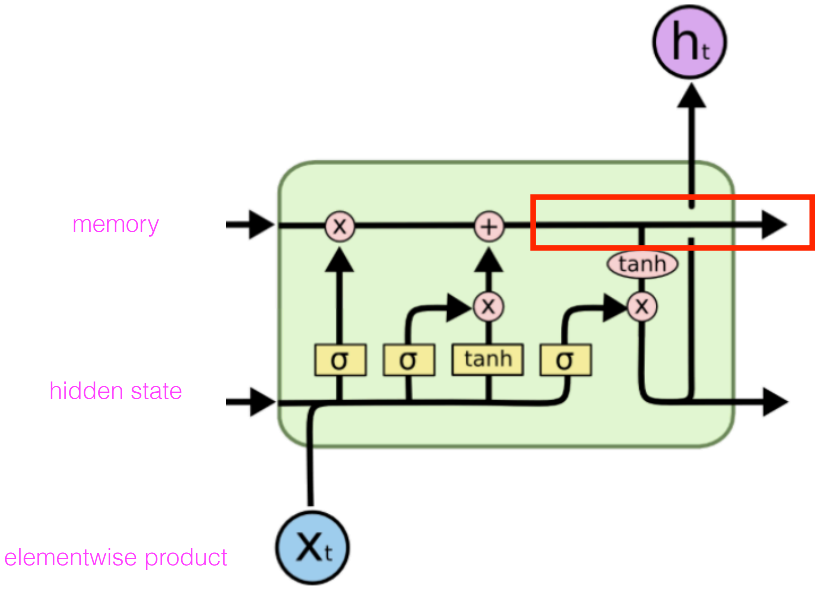
Input gate (but forget some information about the current observation)



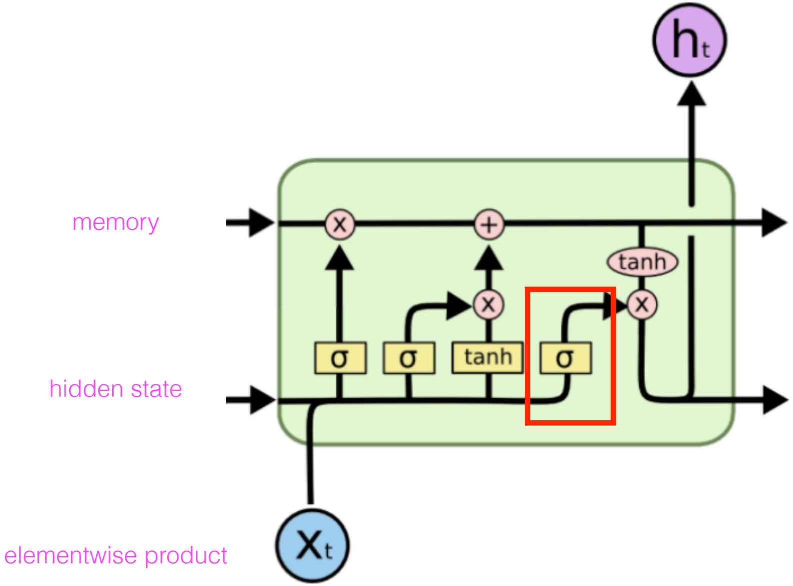
Update the memory (but forget some information about the current observation)



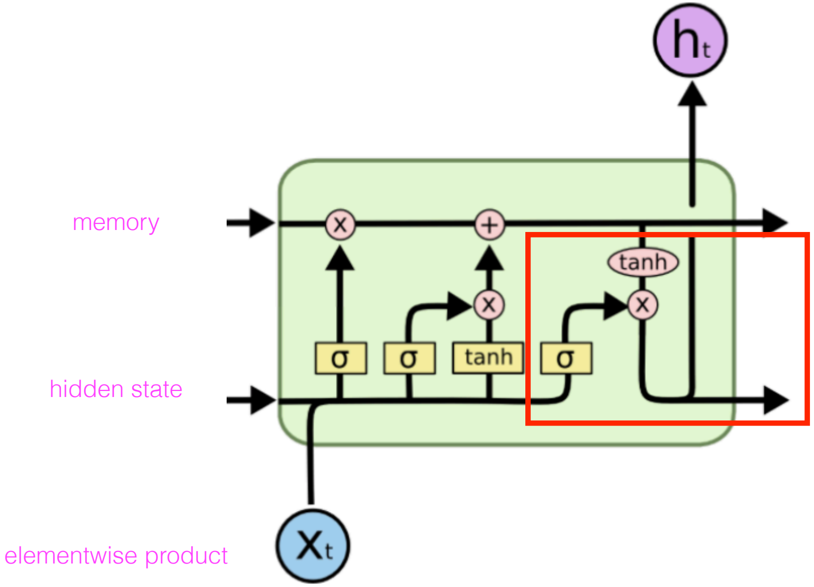
The memory passes directly to the next state



Output gate: forget some information to send to the hidden state

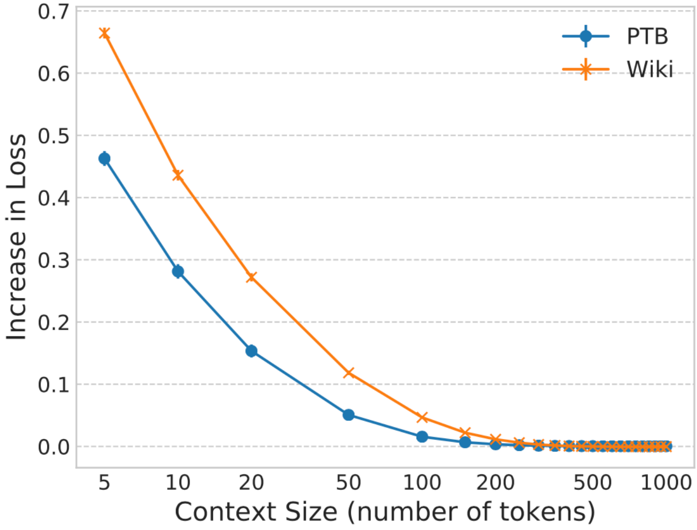


The hidden state is updated with the current observation and new context.



How much context?

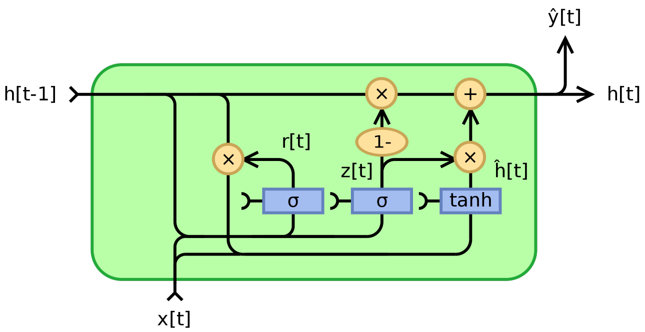
- For language modeling, LSTMs are aware of about 200 words of context
- Ignores word order beyond 50 words



Khandelwal et al. (2018), “Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context” (ACL)

GRU

- A gated recurrent unit adopts the same gating mechanism as an LSTM, but reduces the number of parameters to learn.



- Only one context vector (not a separate memory and hidden state vector) gets passed between timesteps.

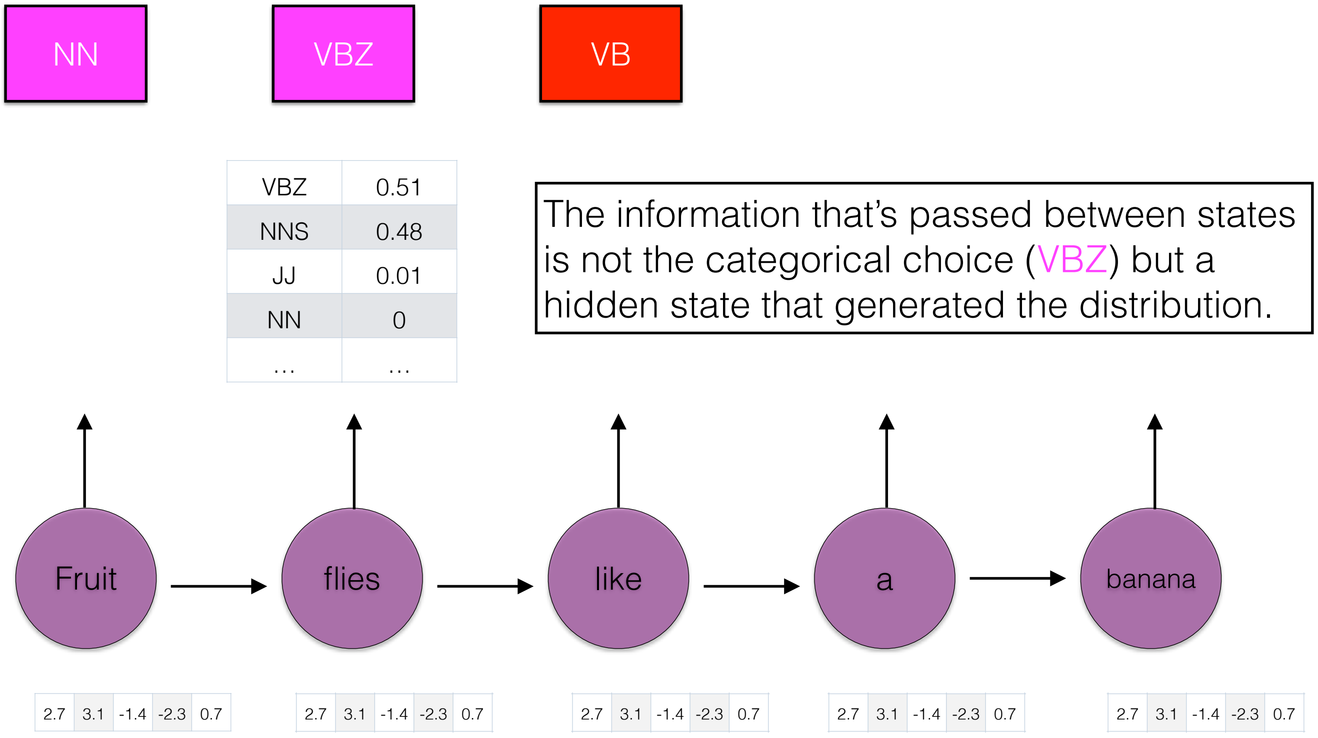
- 2 gates (reset and update) instead of 3.

LSTM/RNN

- Is an RNN the same kind of sequence labeling model as an HMM or MEMM?
- It doesn't use nearby labels in making predictions! (More like logistic regression in this respect)

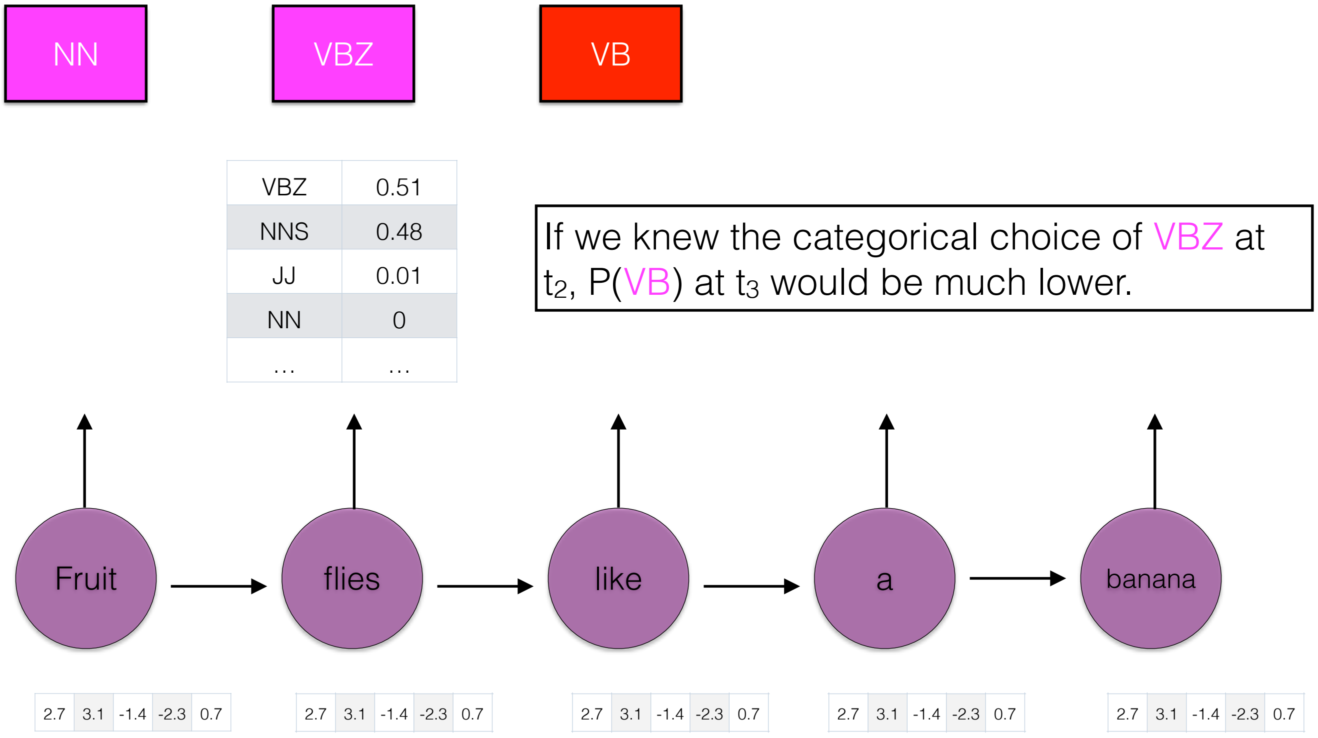
Sequence labeling models

model	form	label dependency	rich features?
Hidden Markov Models	$\prod_{i=1}^{N} P(x_i \mid y_i)\, P(y_i \mid y_{i-1})$	Markov assumption	no
MEMM	$\prod_{i=1}^{N} P(y_i \mid y_{i-1}, x, \beta)$	Markov assumption	yes
RNN	$\prod_{i=1}^{N} P(y_i \mid x_{1:i}, \beta)$	none	distributed



VBZ	0.51
NNS	0.48
JJ	0.01
NN	0
…	…

The information that's passed between states is not the categorical choice (VBZ) but a hidden state that generated the distribution.



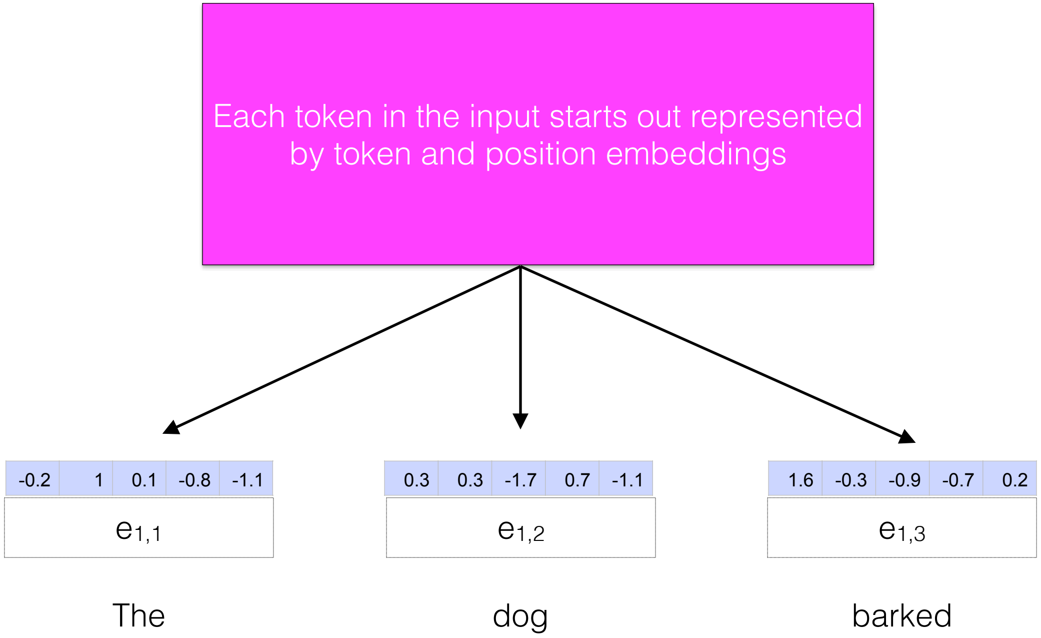
VBZ	0.51
NNS	0.48
JJ	0.01
NN	0
…	…

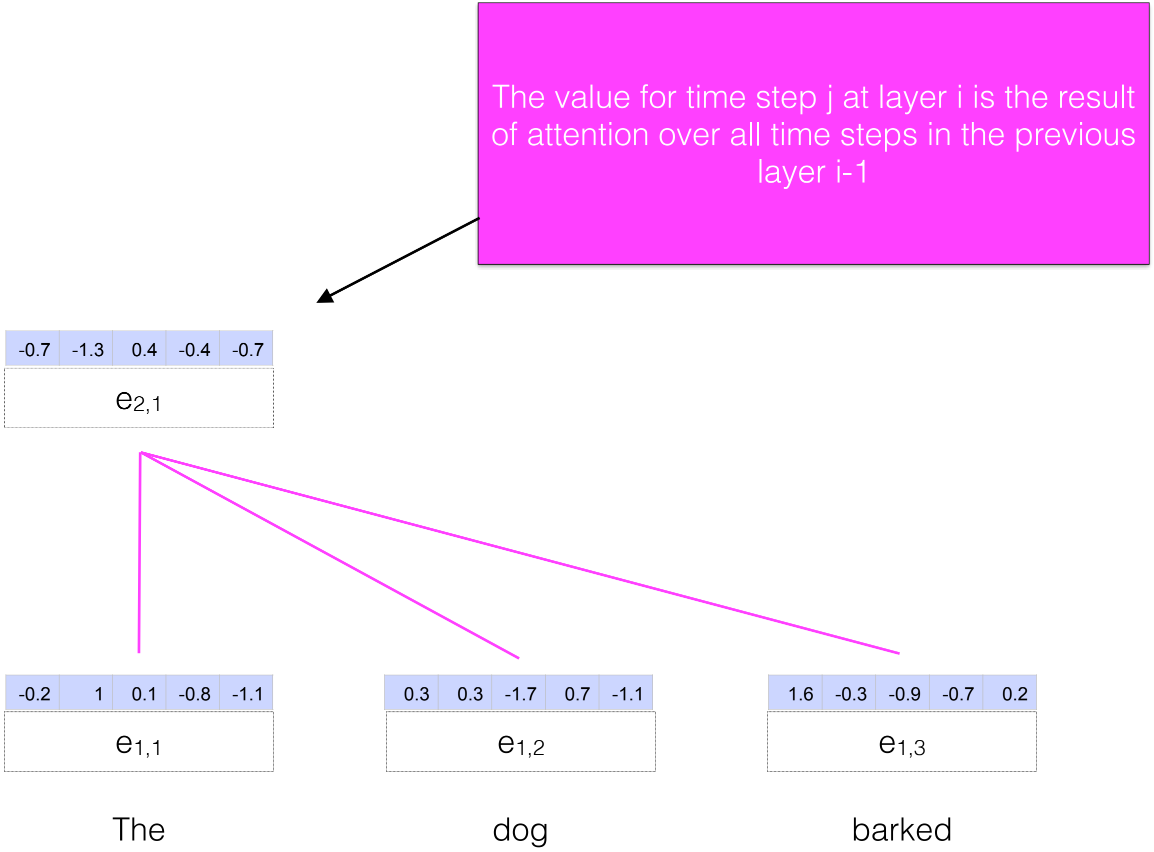
If we knew the categorical choice of VBZ at t_2, P(VB) at t_3 would be much lower.

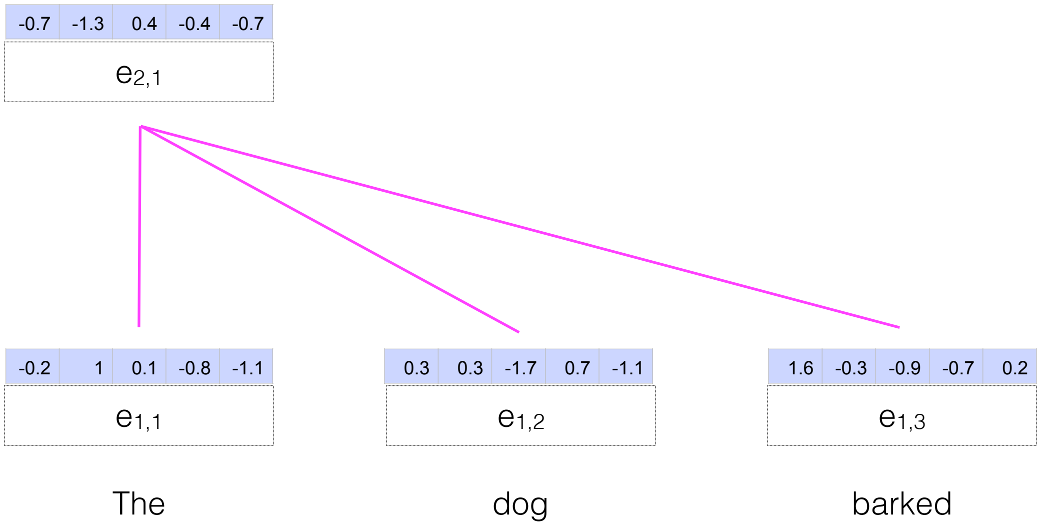
BERT

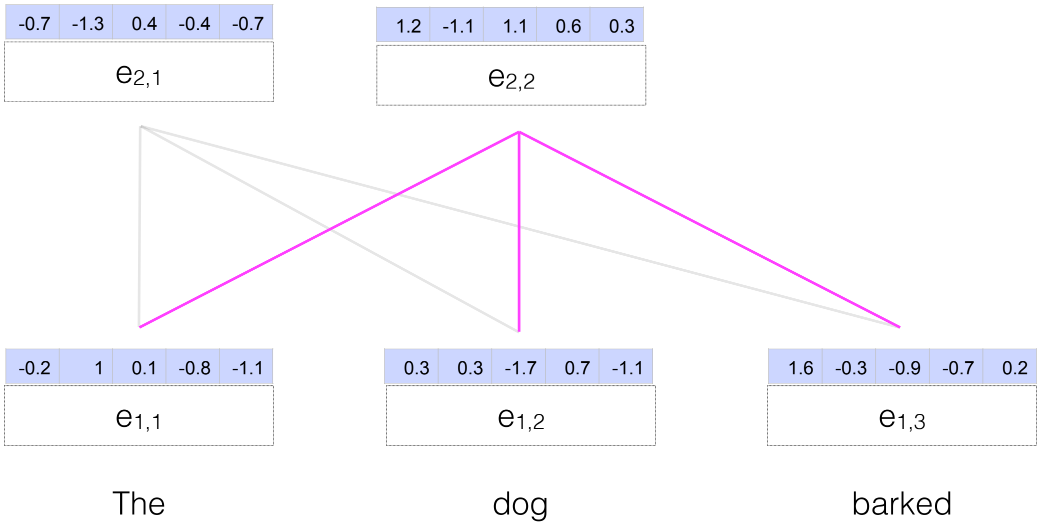
- Transformer-based model (Vaswani et al. 2017) to predict masked word using bidirectional context + next sentence prediction.

- Generates multiple layers of representations for each token sensitive to its context of use.

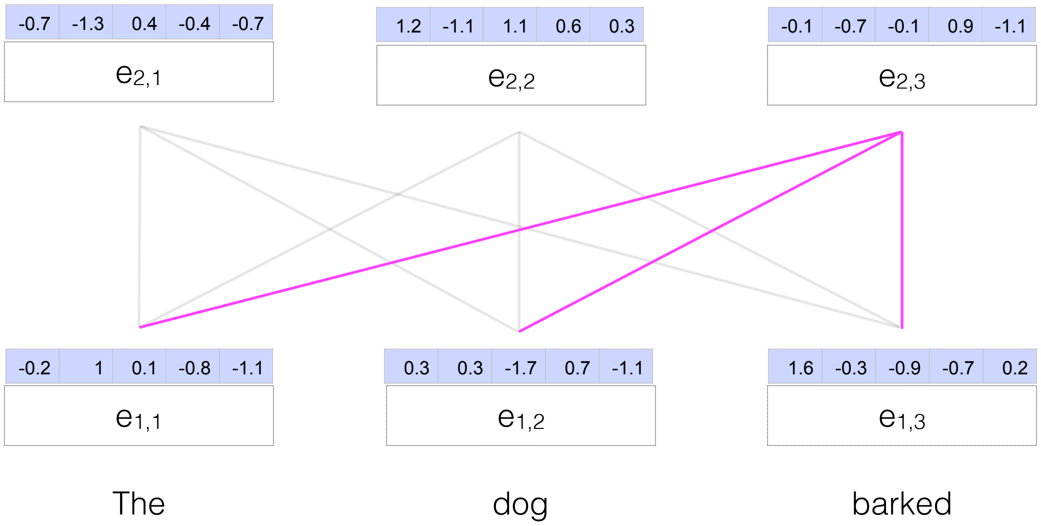

Each token in the input starts out represented by token and position embeddings
-0.2 1 0.1 -0.8 -1.1
e1,1
0.3 0.3 -1.7 0.7 -1.1
e1,2
1.6 -0.3 -0.9 -0.7 0.2
e1,3
The
dog
barked


The value for time step j at layer i is the result of attention over all time steps in the previous layer i-1
-0.7 -1.3 0.4 -0.4 -0.7
e2,1
-0.2 1 0.1 -0.8 -1.1
e1,1
0.3 0.3 -1.7 0.7 -1.1
e1,2
1.6 -0.3 -0.9 -0.7 0.2
e1,3
The
dog
barked

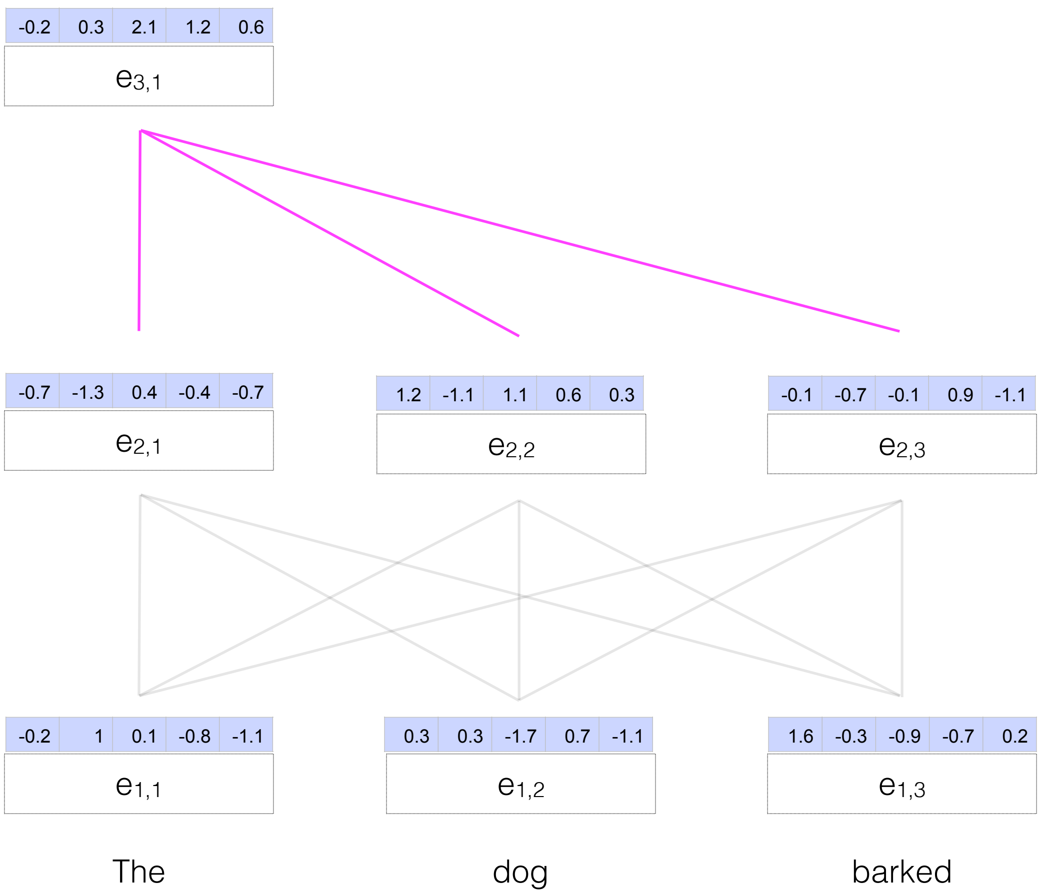

-0.7 -1.3 0.4 -0.4 -0.7
e2,1
-0.2 1 0.1 -0.8 -1.1
e1,1
0.3 0.3 -1.7 0.7 -1.1
e1,2
1.6 -0.3 -0.9 -0.7 0.2
e1,3
The
dog
barked



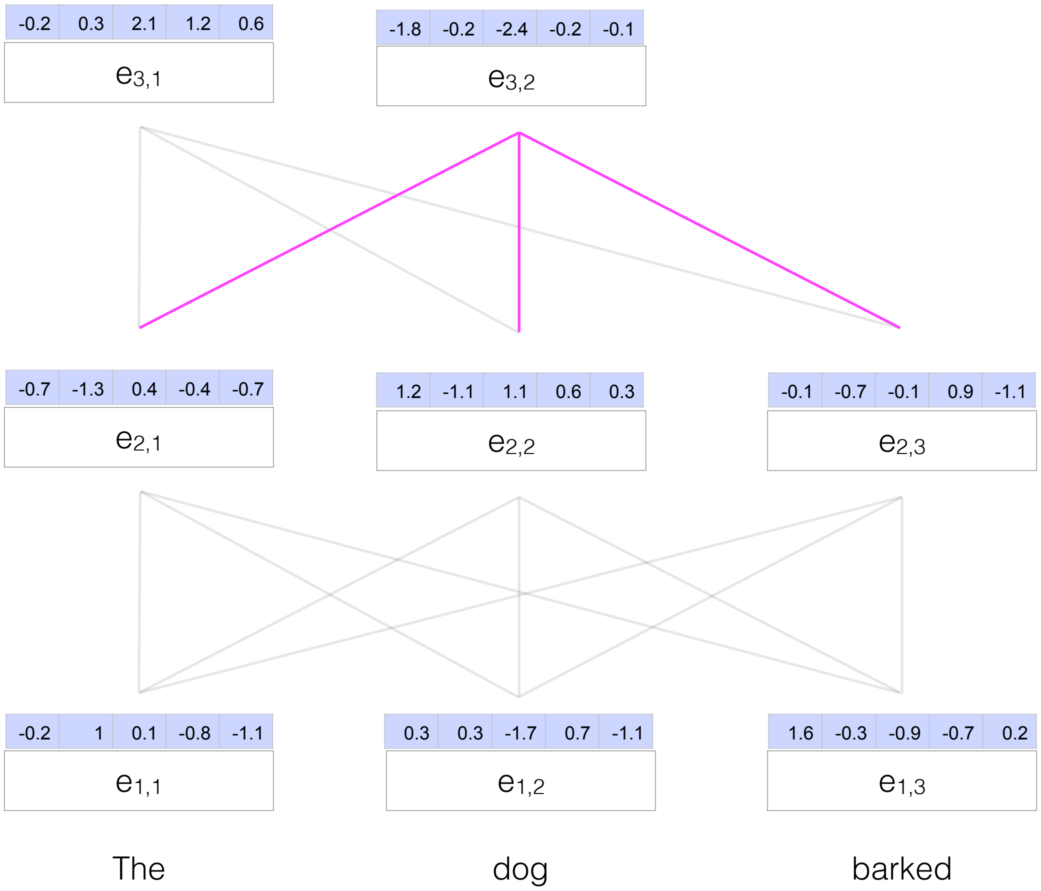
-0.7 -1.3 0.4 -0.4 -0.7
e2,1
1.2 -1.1 1.1 0.6 0.3
e2,2
-0.2 1 0.1 -0.8 -1.1
e1,1
0.3 0.3 -1.7 0.7 -1.1
e1,2
1.6 -0.3 -0.9 -0.7 0.2
e1,3
The
dog
barked

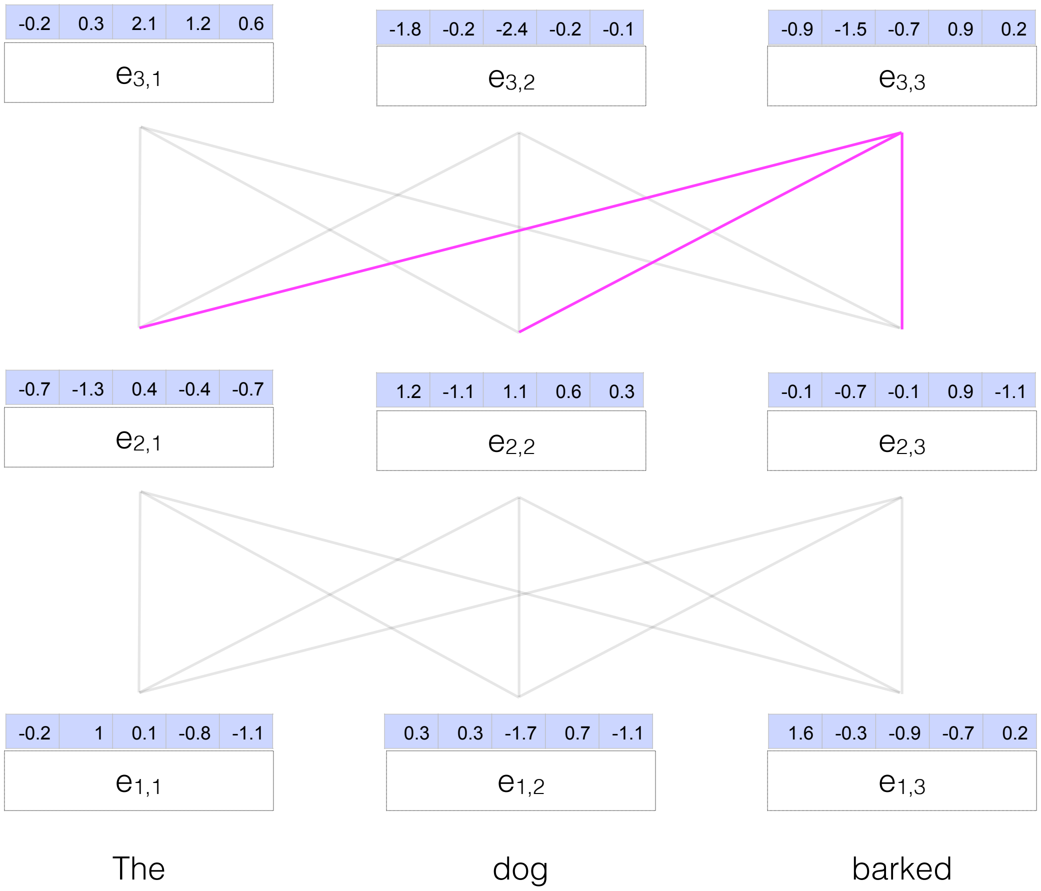


-0.7 -1.3 0.4 -0.4 -0.7
$e_{2,1}$
1.2 -1.1 1.1 0.6 0.3
$e_{2,2}$
-0.1 -0.7 -0.1 0.9 -1.1
$e_{2,3}$
-0.2 1 0.1 -0.8 -1.1
$e_{1,1}$
0.3 0.3 -1.7 0.7 -1.1
$e_{1,2}$
1.6 -0.3 -0.9 -0.7 0.2
$e_{1,3}$
The
dog
barked



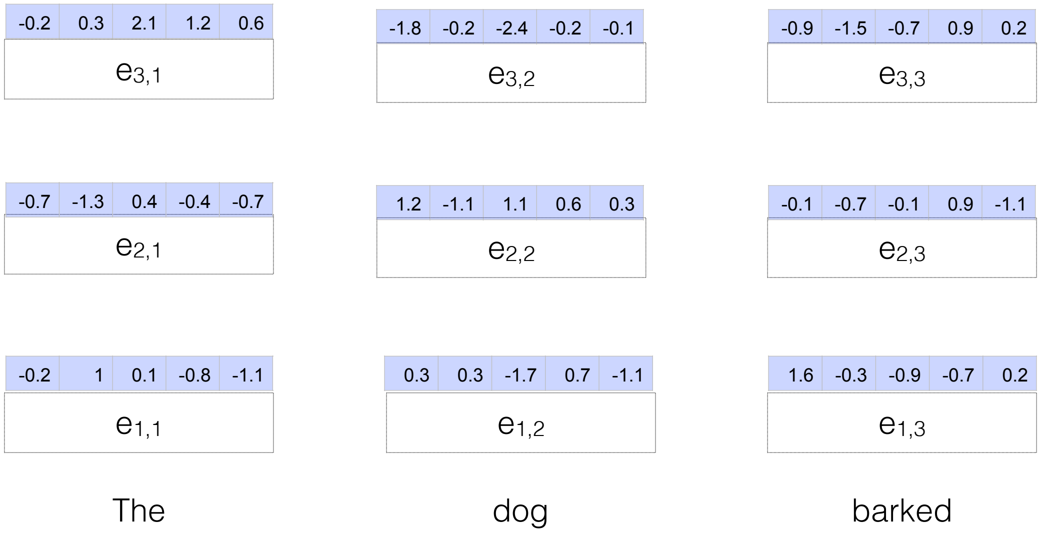
-0.2 0.3 2.1 1.2 0.6
e3,1
-0.7 -1.3 0.4 -0.4 -0.7
e2,1
1.2 -1.1 1.1 0.6 0.3
e2,2
-0.1 -0.7 -0.1 0.9 -1.1
e2,3
-0.2 1 0.1 -0.8 -1.1
e1,1
0.3 0.3 -1.7 0.7 -1.1
e1,2
1.6 -0.3 -0.9 -0.7 0.2
e1,3
The
dog
barked


-0.2 0.3 2.1 1.2 0.6
$e_{3,1}$
-1.8 -0.2 -2.4 -0.2 -0.1
$e_{3,2}$
-0.7 -1.3 0.4 -0.4 -0.7
$e_{2,1}$
1.2 -1.1 1.1 0.6 0.3
$e_{2,2}$
-0.1 -0.7 -0.1 0.9 -1.1
$e_{2,3}$
-0.2 1 0.1 -0.8 -1.1
$e_{1,1}$
0.3 0.3 -1.7 0.7 -1.1
$e_{1,2}$
1.6 -0.3 -0.9 -0.7 0.2
$e_{1,3}$
The
dog
barked



-0.2 0.3 2.1 1.2 0.6
$e_{3,1}$
-1.8 -0.2 -2.4 -0.2 -0.1
$e_{3,2}$
-0.9 -1.5 -0.7 0.9 0.2
$e_{3,3}$
-0.7 -1.3 0.4 -0.4 -0.7
$e_{2,1}$
1.2 -1.1 1.1 0.6 0.3
$e_{2,2}$
-0.1 -0.7 -0.1 0.9 -1.1
$e_{2,3}$
-0.2 1 0.1 -0.8 -1.1
$e_{1,1}$
0.3 0.3 -1.7 0.7 -1.1
$e_{1,2}$
1.6 -0.3 -0.9 -0.7 0.2
$e_{1,3}$
The
dog
barked

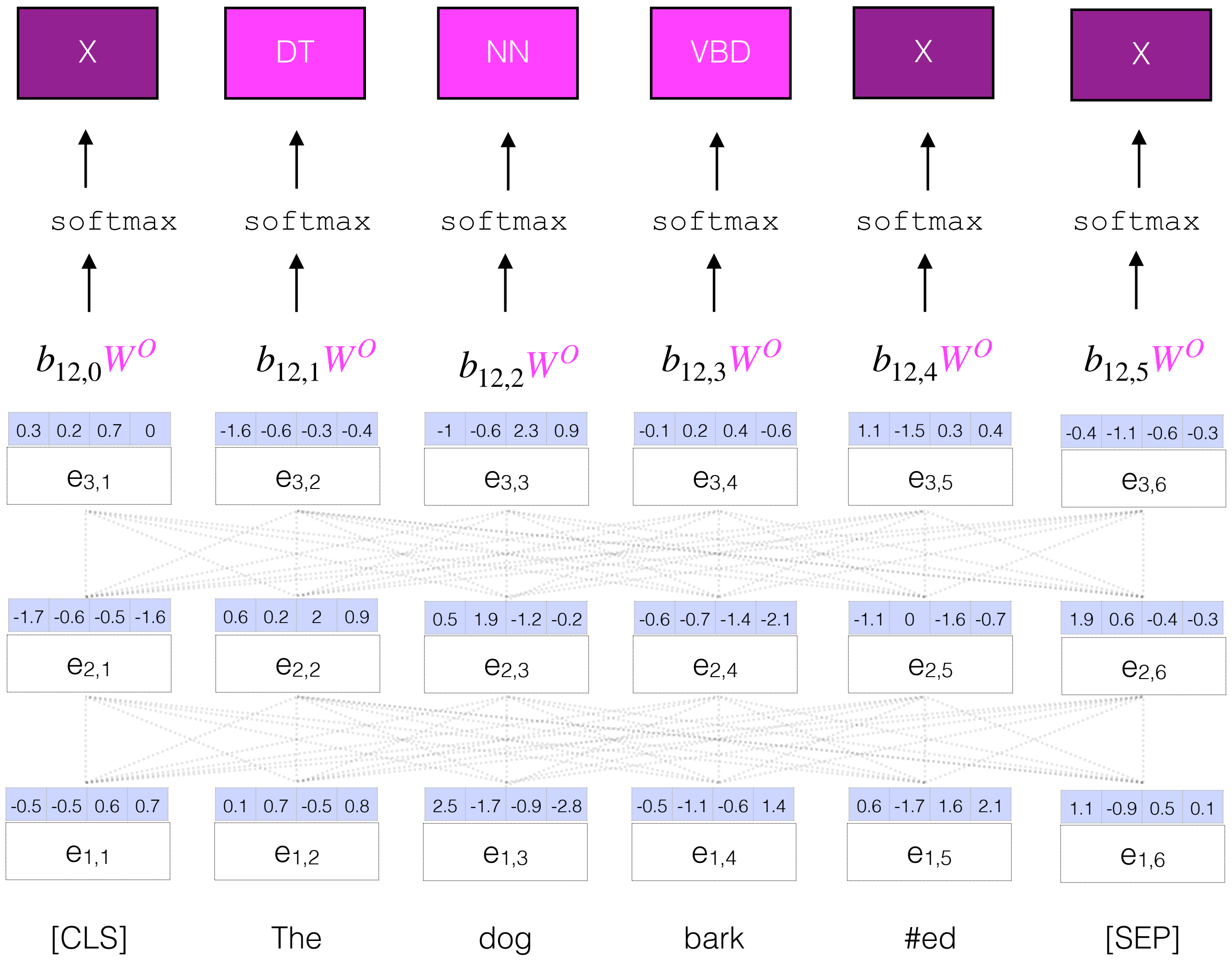
At the end of this process, we have one representation for each layer for each token



BERT

- BERT can be used not only as a language model to generate contextualized word representations, but also as a predictive model whose parameters are fine-tuned to a task.

neutral
softmax
$b_{12,0}W^O$
0.3 0.2 0.7 0
-1.6 -0.6 -0.3 -0.4
-1 -0.6 2.3 0.9
-0.1 0.2 0.4 -0.6
1.1 -1.5 0.3 0.4
-0.4 -1.1 -0.6 -0.3
e3,1
e3,2
e3,3
e3,4
e3,5
e3,6
-1.7 -0.6 -0.5 -1.6
0.6 0.2 2 0.9
0.5 1.9 -1.2 -0.2
-0.6 -0.7 -1.4 -2.1
-1.1 0 -1.6 -0.7
1.9 0.6 -0.4 -0.3
e2,1
e2,2
e2,3
e2,4
e2,5
e2,6
-0.5 -0.5 0.6 0.7
0.1 0.7 -0.5 0.8
2.5 -1.7 -0.9 -2.8
-0.5 -1.1 -0.6 1.4
0.6 -1.7 1.6 2.1
1.1 -0.9 0.5 0.1
e1,1
e1,2
e1,3
e1,4
e1,5
e1,6
[CLS]
The
dog
bark
#ed
[SEP]


X
DT
NN
VBD
X
X
softmax
softmax
softmax
softmax
softmax
softmax
$b_{12,0}W^O$
$b_{12,1}W^O$
$b_{12,2}W^O$
$b_{12,3}W^O$
$b_{12,4}W^O$
$b_{12,5}W^O$
0.3 0.2 0.7 0
-1.6 -0.6 -0.3 -0.4
-1 -0.6 2.3 0.9
-0.1 0.2 0.4 -0.6
1.1 -1.5 0.3 0.4
-0.4 -1.1 -0.6 -0.3
$e_{3,1}$
$e_{3,2}$
$e_{3,3}$
$e_{3,4}$
$e_{3,5}$
$e_{3,6}$
-1.7 -0.6 -0.5 -1.6
0.6 0.2 2 0.9
0.5 1.9 -1.2 -0.2
-0.6 -0.7 -1.4 -2.1
-1.1 0 -1.6 -0.7
1.9 0.6 -0.4 -0.3
$e_{2,1}$
$e_{2,2}$
$e_{2,3}$
$e_{2,4}$
$e_{2,5}$
$e_{2,6}$
-0.5 -0.5 0.6 0.7
0.1 0.7 -0.5 0.8
2.5 -1.7 -0.9 -2.8
-0.5 -1.1 -0.6 1.4
0.6 -1.7 1.6 2.1
1.1 -0.9 0.5 0.1
$e_{1,1}$
$e_{1,2}$
$e_{1,3}$
$e_{1,4}$
$e_{1,5}$
$e_{1,6}$
[CLS]
The
dog
bark
#ed
[SEP]

BERT

- Pre-training: train BERT through masked language modeling and next-sentence prediction to learn the parameters of BERT layers. Trained on Wikipedia + BookCorpus.

- Task fine-tuning: add additional linear transformation + softmax to get distribution over output space. Trained on annotated data.

Logistics

- Exam1 grading is being finalized.
- No homework this week
    - Homework 4 will be released by Friday.
- AP1 is due this Sunday March 3
- Quiz 4 will be out this Friday afternoon (due Monday).
- Next Week: Parsing