



Natural Language Processing

Info 159/259

Midterm review (Mar 8, 2022)

David Bamman, UC Berkeley

Midterm

- Thursday 3/10, 2:10-3:30pm PST on bCourses. This midterm must be carried out entirely **independently!**
- You are free to use any of your notes, readings, or lecture material; the midterm will cover all material through 3/3 lecture on parsing.
- Questions will resemble what you have already experienced with the quizzes (e.g., multiple choice) along with some numerical answer/fill-in-the-blank questions as well. Have your calculator ready!
- You can expect more attention to the topics we cover in lectures and in the homeworks, but everything in the lectures and readings is fair game.

Big ideas

- Classification
 - Logistic regression, feedforward neural networks, CNN, BERT, Attention, transformers
- Where does NLP data come from?
 - Annotation process
 - Interannotator agreement
- Language modeling
 - Markov assumption, featurized, neural, prompting
 - Evaluation metrics (accuracy, precision, recall, F score, perplexity, span F1, parseval)

Big ideas

- Lexical semantics and word representations
 - Distributional hypothesis
 - Distributed representations
 - Subword embedding models
 - Contextualized word representations (ELMO/BERT)
- Sequence labeling
 - POS, NER
 - Methods: HMM, MEMM, CRF, RNN, BiRNN, LSTM, BERT
- Trees
 - Phrase-structure parsing, CFG, PCFG
 - CKY for recognition, parsing

Big ideas

- What defines the models we've seen so far? What formally distinguishes an HMM from an MEMM? How do we train those models?
- For all of the problems we've seen (sentiment analysis, POS tagging, phrase structure parsing), how do we evaluate the performance of different models?
- If faced with a new NLP problem, how would you decide between the alternatives you know about? How would you adapt an MEMM, for example, to a new problem?

to

	VB	DT	NN	END
START	0.9	0.1	0	
VB	0.1	0.6	0.1	0.2
DT	0	0.1	0.9	0
NN	0.1	0.1	0.1	0.7

	book	the	flight	a
VB	0.9	0.03	0.04	0.03
DT	0	0.6	0	0.4
NN	0.5	0	0.5	0

END				
VB				
DT				
NN				
START				

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB				
DT				
NN				
START				

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	$\log P(\text{VB} \text{START}) + \log P(\text{book} \text{VB})$ $0.11 - 0.11 = -0.22$			
DT	$\log P(\text{DT} \text{START}) + \log P(\text{book} \text{DT})$ $-2.30 - \infty = -\infty$			
NN	$\log P(\text{NN} \text{START}) + \log P(\text{book} \text{NN})$ $-\infty - 0.69 = -\infty$			
START				

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	$\log P(\text{VB} \text{START}) + \log P(\text{book} \text{VB})$ $0.11 - 0.11 = -0.22$			
DT	$\log P(\text{DT} \text{START}) + \log P(\text{book} \text{DT})$ $-2.30 - \infty = -\infty$			
NN	$\log P(\text{NN} \text{START}) + \log P(\text{book} \text{NN})$ $-\infty - 0.69 = -\infty$			
START				



Book	the	flight
------	-----	--------

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22			
DT	$-\infty$			
NN	$-\infty$			
START				

Max:

- $\log P(\text{VB}|\text{VB}) + \log P(\text{the}|\text{VB}) - 0.22$
- $\log P(\text{VB}|\text{DT}) + \log P(\text{the}|\text{VB}) - \infty$
- $\log P(\text{VB}|\text{NN}) + \log P(\text{the}|\text{VB}) - \infty$

Book

the

flight



from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03		
DT	$-\infty$			
NN	$-\infty$			
START				

Max:

- $-2.30 + -3.51 - 0.22 = -6.03$
- $-\infty$
- $-\infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03		
DT	$-\infty$			
NN	$-\infty$			
START				

Max:

- $\log P(\text{DT}|\text{VB}) + \log P(\text{the}|\text{DT}) - 0.22$
- $\log P(\text{DT}|\text{DT}) + \log P(\text{the}|\text{DT}) - \infty$
- $\log P(\text{DT}|\text{NN}) + \log P(\text{the}|\text{DT}) - \infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03		
DT	$-\infty$	-1.24		
NN	$-\infty$			
START				

Max:

- $-0.51 + -0.51 - 0.22 = -1.24$
- $-\infty$
- $-\infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03		
DT	$-\infty$	-1.24		
NN	$-\infty$			
START				

Max:

- $\log P(\text{NN}|\text{VB}) + \log P(\text{the}|\text{NN}) - 0.22$
- $\log P(\text{NN}|\text{DT}) + \log P(\text{the}|\text{NN}) - \infty$
- $\log P(\text{NN}|\text{NN}) + \log P(\text{the}|\text{NN}) - \infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03		
DT	$-\infty$	-1.24		
NN	$-\infty$	$-\infty$		
START				

Max:

- $-2.30 + -\infty - 0.22$
- $\log P(\text{NN}|\text{DT}) + \log P(\text{the}|\text{NN}) -\infty$
- $\log P(\text{NN}|\text{NN}) + \log P(\text{the}|\text{NN}) -\infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03		
DT	$-\infty$	-1.24		
NN	$-\infty$	$-\infty$		
START				

Max:

- $\log P(\text{VB}|\text{VB}) + \log P(\text{flight}|\text{VB}) - 6.03$
- $\log P(\text{VB}|\text{DT}) + \log P(\text{flight}|\text{VB}) - 1.24$
- $\log P(\text{VB}|\text{NN}) + \log P(\text{flight}|\text{VB}) - \infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03	-11.55	
DT	$-\infty$	-1.24		
NN	$-\infty$	$-\infty$		
START				

Max:

- $-2.30 - 3.22 - 6.03 = -11.55$
- $-\infty + -3.22 - 1.24$
- $-2.30 - 3.22 - \infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03	-11.55	
DT	$-\infty$	-1.24		
NN	$-\infty$	$-\infty$		
START				

Max:

- $\log P(\text{DT}|\text{VB}) + \log P(\text{flight}|\text{DT}) - 6.03$
- $\log P(\text{DT}|\text{DT}) + \log P(\text{flight}|\text{DT}) - 1.24$
- $\log P(\text{DT}|\text{NN}) + \log P(\text{flight}|\text{DT}) - \infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03	-11.55	
DT	$-\infty$	-1.24	$-\infty$	
NN	$-\infty$	$-\infty$		
START				

Max:

- $-0.51 - \infty - 6.03$
- $\log P(\text{DT}|\text{DT}) - \infty - 1.24$
- $\log P(\text{DT}|\text{NN}) - \infty - \infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03	-11.55	
DT	$-\infty$	-1.24	$-\infty$	
NN	$-\infty$	$-\infty$		
START				

Max:

- $\log P(\text{NN}|\text{VB}) + \log P(\text{flight}|\text{NN}) - 6.03$
- $\log P(\text{NN}|\text{DT}) + \log P(\text{flight}|\text{NN}) - 1.24$
- $\log P(\text{NN}|\text{NN}) + \log P(\text{flight}|\text{NN}) - \infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

	book	the	flight	a
VB	-0.11	-3.51	-3.22	-3.51
DT	$-\infty$	-0.51	$-\infty$	-0.92
NN	-0.69	$-\infty$	-0.69	$-\infty$

END				
VB	-0.22	-6.03	-11.55	
DT	$-\infty$	-1.24	$-\infty$	
NN	$-\infty$	$-\infty$	-2.04	
START				

Max:

- $-2.30 + -0.69 - 6.03 = -9.02$
- $-0.11 - 0.69 - 1.24 = -2.04$
- $-2.30 - 0.69 - \infty$

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

Max:

- $\log P(\text{END}|\text{VB}) - 11.55$
- $\log P(\text{END}|\text{DT}) - \infty$
- $\log P(\text{END}|\text{NN}) - 2.04$

END				
VB	-0.22	-6.03	-11.55	
DT	$-\infty$	-1.24	$-\infty$	
NN	$-\infty$	$-\infty$	-2.04	
START				

Book

the

flight

from

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

from

Max:

- $-1.61 - 11.55 = -13.16$
- $\log P(\text{END}|\text{DT}) - \infty$
- $-0.36 - 2.04 = -2.40$

END				-2.40
VB	-0.22	-6.03	-11.55	
DT	$-\infty$	-1.24	$-\infty$	
NN	$-\infty$	$-\infty$	-2.04	
START				

Book

the

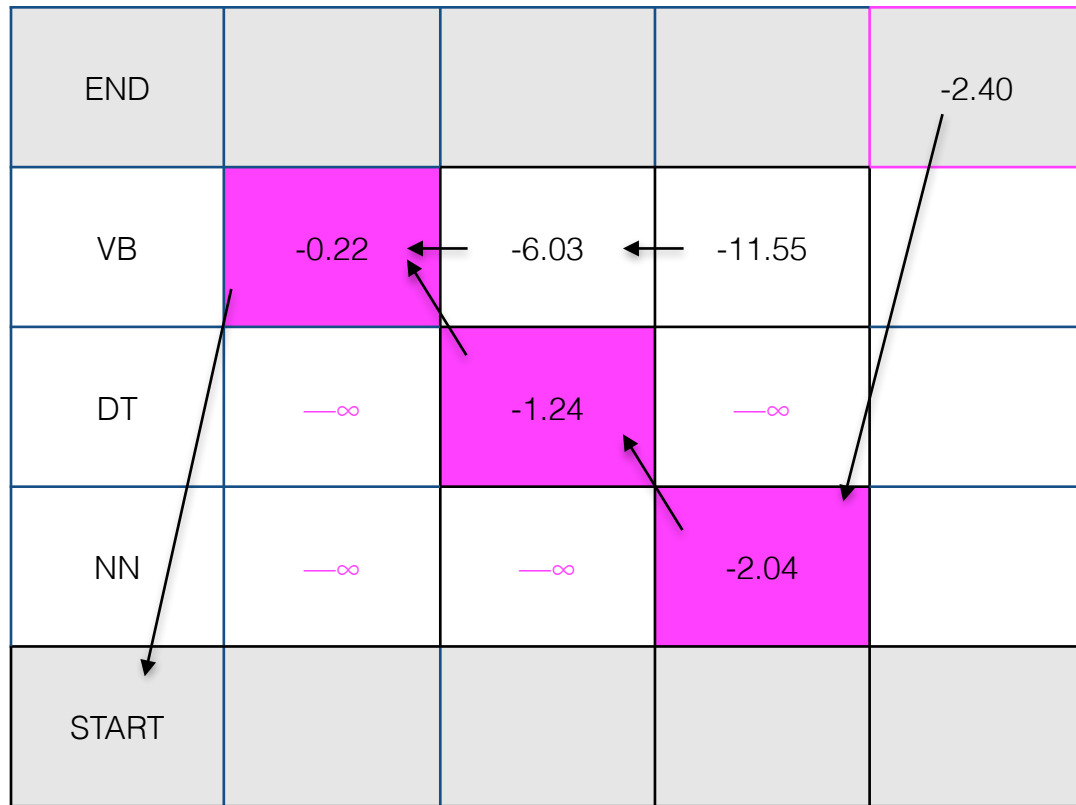
flight

to

	VB	DT	NN	END
START	-0.11	-2.30	$-\infty$	
VB	-2.30	-0.51	-2.30	-1.61
DT	$-\infty$	-2.30	-0.11	$-\infty$
NN	-2.30	-2.30	-2.30	-0.36

Max:

- $-1.61 - 11.55 = -13.16$
- $\log P(\text{END}|\text{DT}) - \infty$
- $-0.36 - 2.04 = -2.40$



Book

the

flight

Can we review how PCFG works? I'm confused on how the PCFG takes in a sentence as input and calculates probabilities as an output.

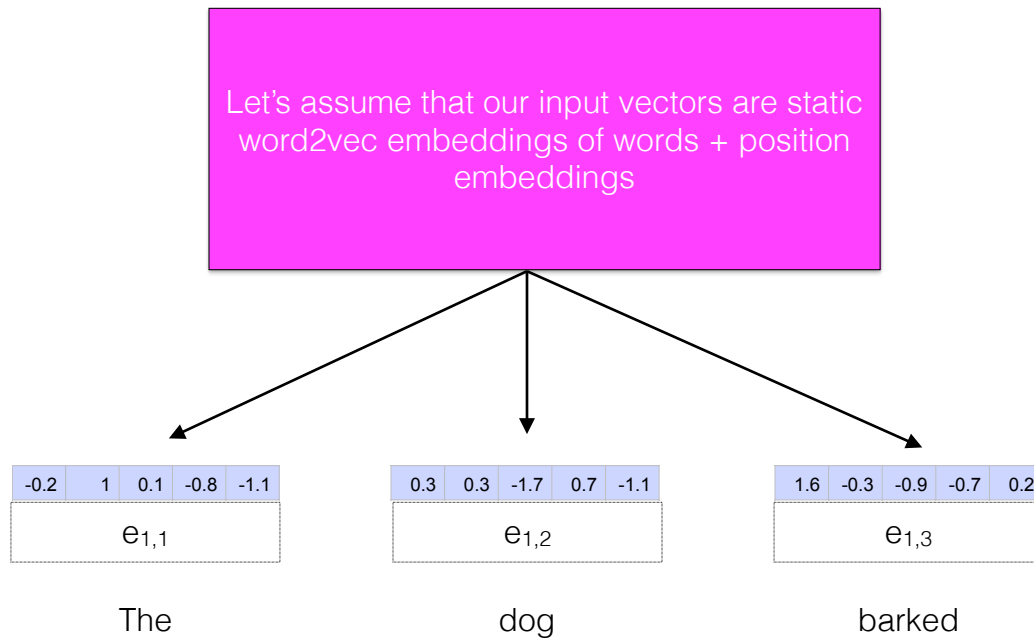
Production	log prob
VB →book	-3.1
NN→book	-5.5
NN→flight	-5.7
DT→the	-2.3
DT→a	-3.1

Production	log prob
NP→DT NN	-1.7
S→VB NP	-1.3
S→VP NP	-2.3
NP→NN NN	-4.3

Book	the	flight

Can you explain transformers again. Preferably, with an example.

Self-Attention



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
$e_{2,1}$				

-0.2	1	0.1	-0.8	-1.1
$e_{1,1}$				

The

0.3	0.3	-1.7	0.7	-1.1
$e_{1,2}$				

dog

1.6	-0.3	-0.9	-0.7	0.2
$e_{1,3}$				

barked

- Let's separate out the different functions that an input vector has in attention by transforming it into separate representations for its role in a weighted sum (the **value**) from the roles used to assess compatibility (the **query** and **key**).

query

$$q_{1,1} \in \mathbb{R}^{37} \quad (e_{1,1} W^Q)$$

key

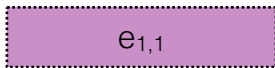
$$k_{1,1} \in \mathbb{R}^{37} \quad (e_{1,1} W^K)$$

value

$$v_{1,1} \in \mathbb{R}^{100} \quad (e_{1,1} W^V)$$

original value

$$e_{1,1} \in \mathbb{R}^{100}$$



The

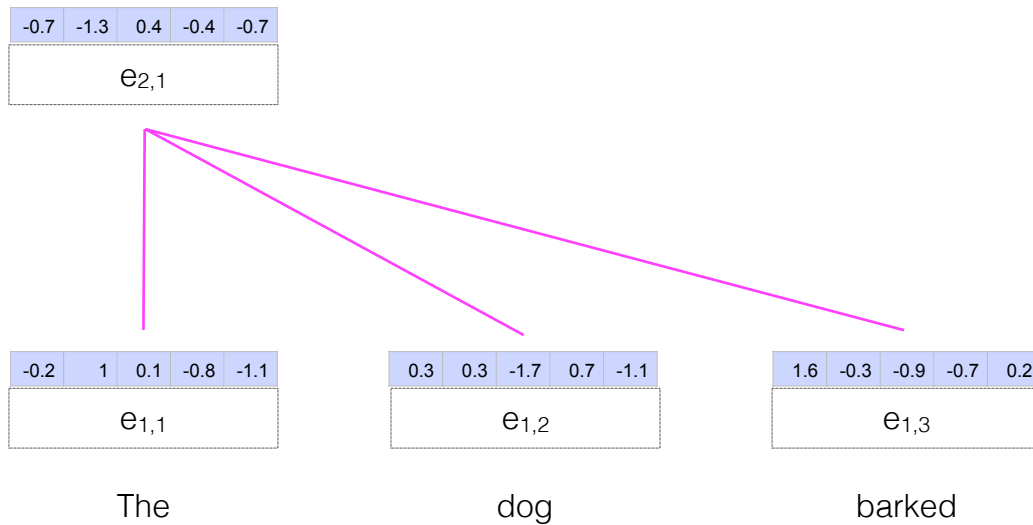
$$W^Q \in \mathbb{R}^{100 \times 37}$$

$$W^K \in \mathbb{R}^{100 \times 37}$$

$$W^V \in \mathbb{R}^{100 \times 100}$$

These are all parameters we *learn*. 100 is the original input dimension; 37 is a hyper-parameter we choose.

Self attention **from** “The” at position 1 to every token in the sentence

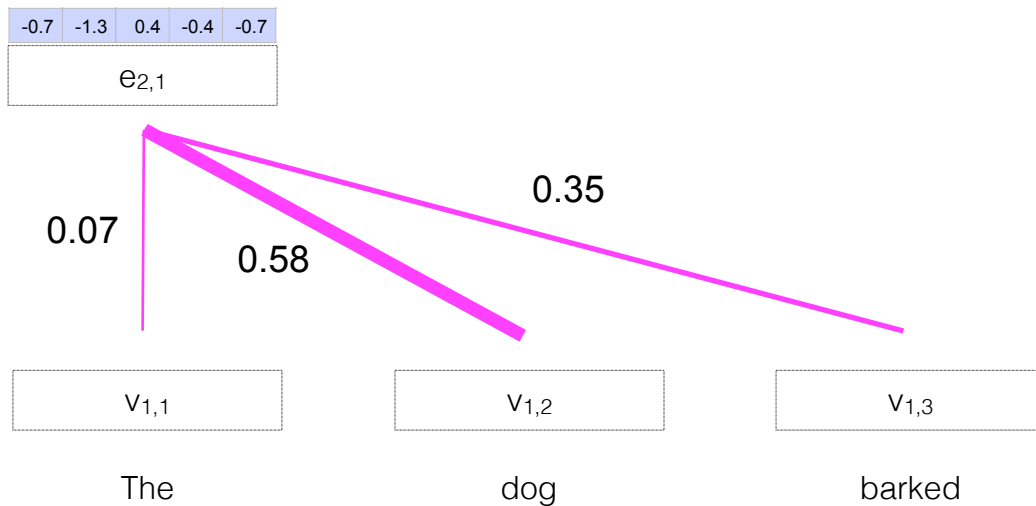


- The compatibility score between two words is the dot product between their respective **query** and **key** vectors.

$$\text{score}(e_i, e_j) = q_i \cdot k_j$$

<i>a</i>	0.07	0.58	0.35	$a = \text{softmax}(\text{scores})$
<i>scores</i>	-1.4	0.64	0.14	
	$q_1 \cdot k_1$	$q_1 \cdot k_2$	$q_1 \cdot k_3$	
	-0.210.1-0.8-1.1 $e_{1,1}$	0.30.3-1.70.7-1.1 $e_{1,2}$	1.6-0.3-0.9-0.70.2 $e_{1,3}$	
	The	dog	barked	

- The output of attention is a weighted sum over the **values** of the previous layer.



How does multihead attention work again?

Multihead attention

- Attention in transformers is essentially a set of learned parameters (W^Q , W^K , W^V) and a mathematical expression for how an input is transformed into an output through operations involving those parameters.

$$Q = XW^Q; K = XW^K; V = XW^V$$

$$\text{SelfAttention}(Q, K, V) = \text{softmax} \left(\frac{QK^\top}{\sqrt{d_k}} \right) V$$

Scaled by the dimensionality of
the key vectors ($\sqrt{d_k}$)

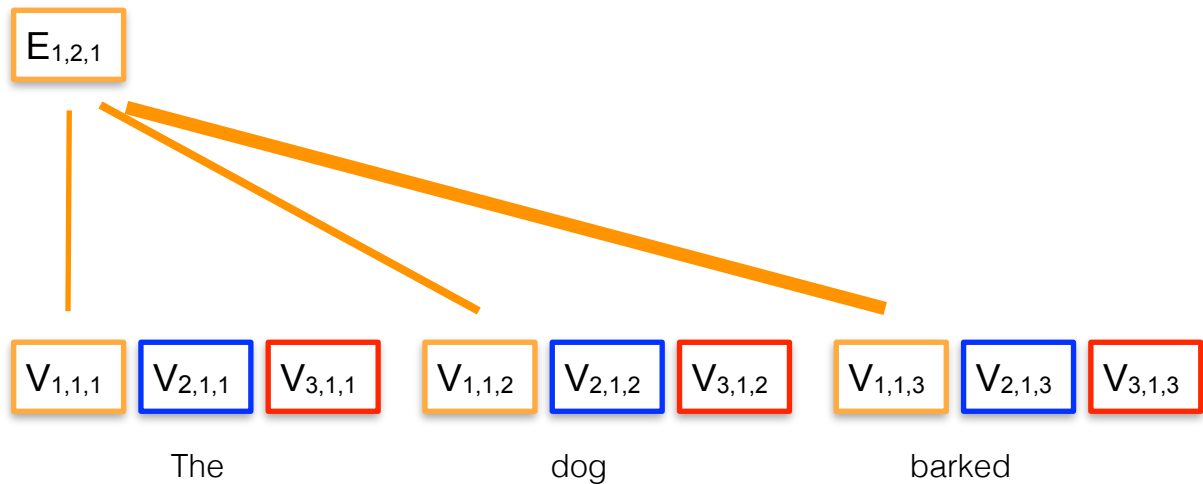
Multihead attention

- Attention provides one view on the data; just like we use multiple filters in CNNs to provide multiple perspectives (by learning separate parameters for each one), so too can we learn multiple perspectives in a transformer by learning multiple (W^Q , W^K , W^V) sets.

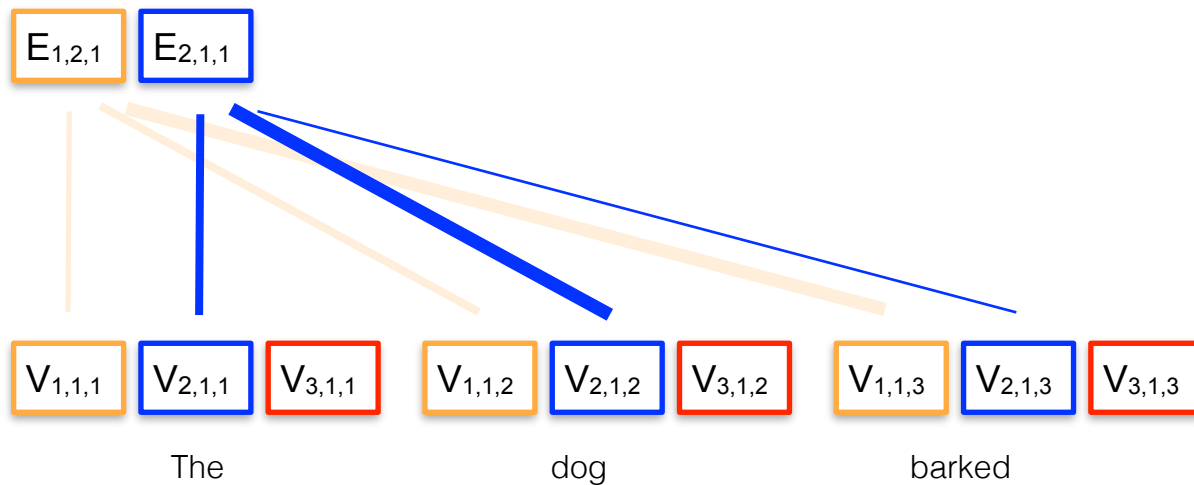
$$Q_i = XW_i^Q; K = XW_i^K; V = XW_i^V$$

$$\text{SelfAttention}(Q_i, K_i, V_i) = \text{softmax}\left(\frac{Q_i K_i^\top}{\sqrt{d_k}}\right) V_i$$

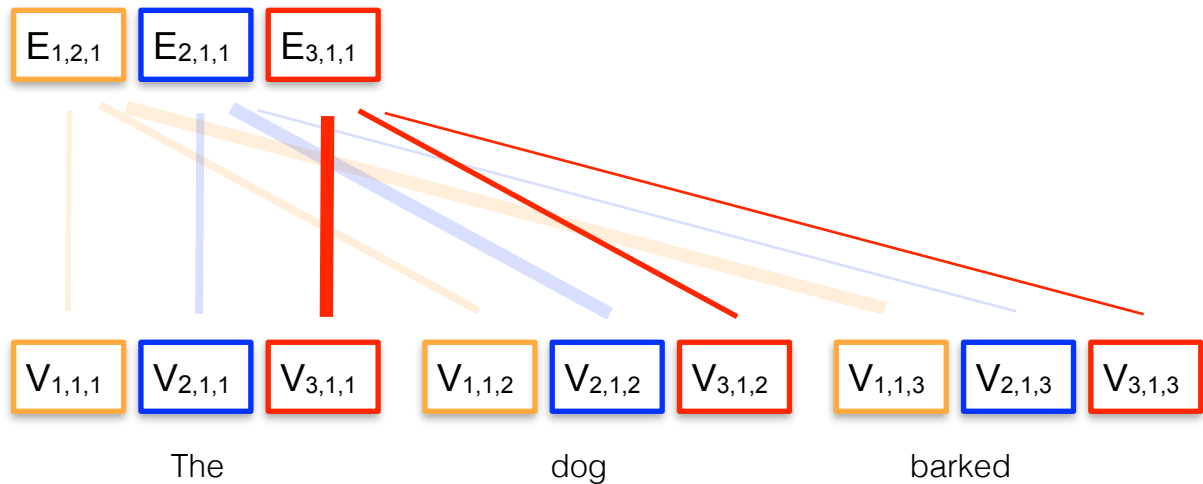
- With multihead attention, each attention head generates its own output vector i based on its own W_i^Q , W_i^K and W_i^V



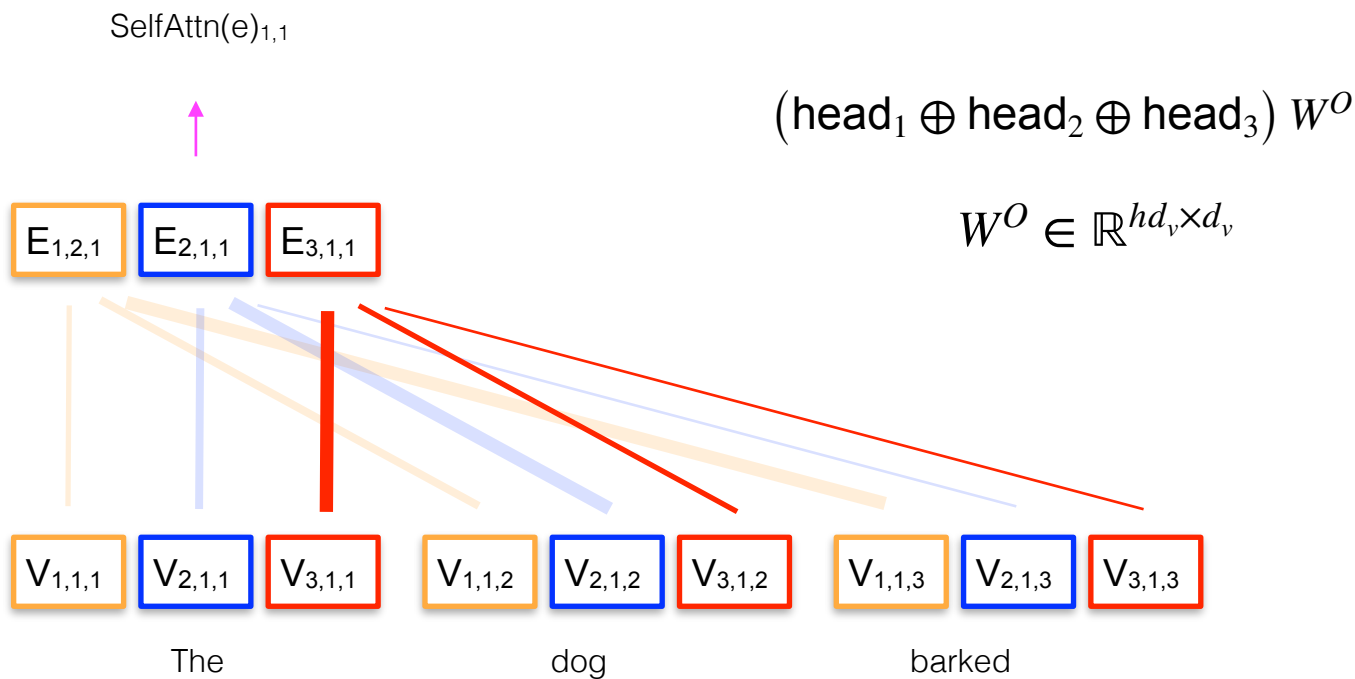
- With multihead attention, each attention head generates its own output vector i based on its own W_i^Q , W_i^K and W_i^V



- With multihead attention, each attention head generates its own output vector i based on its own W_i^Q , W_i^K and W_i^V

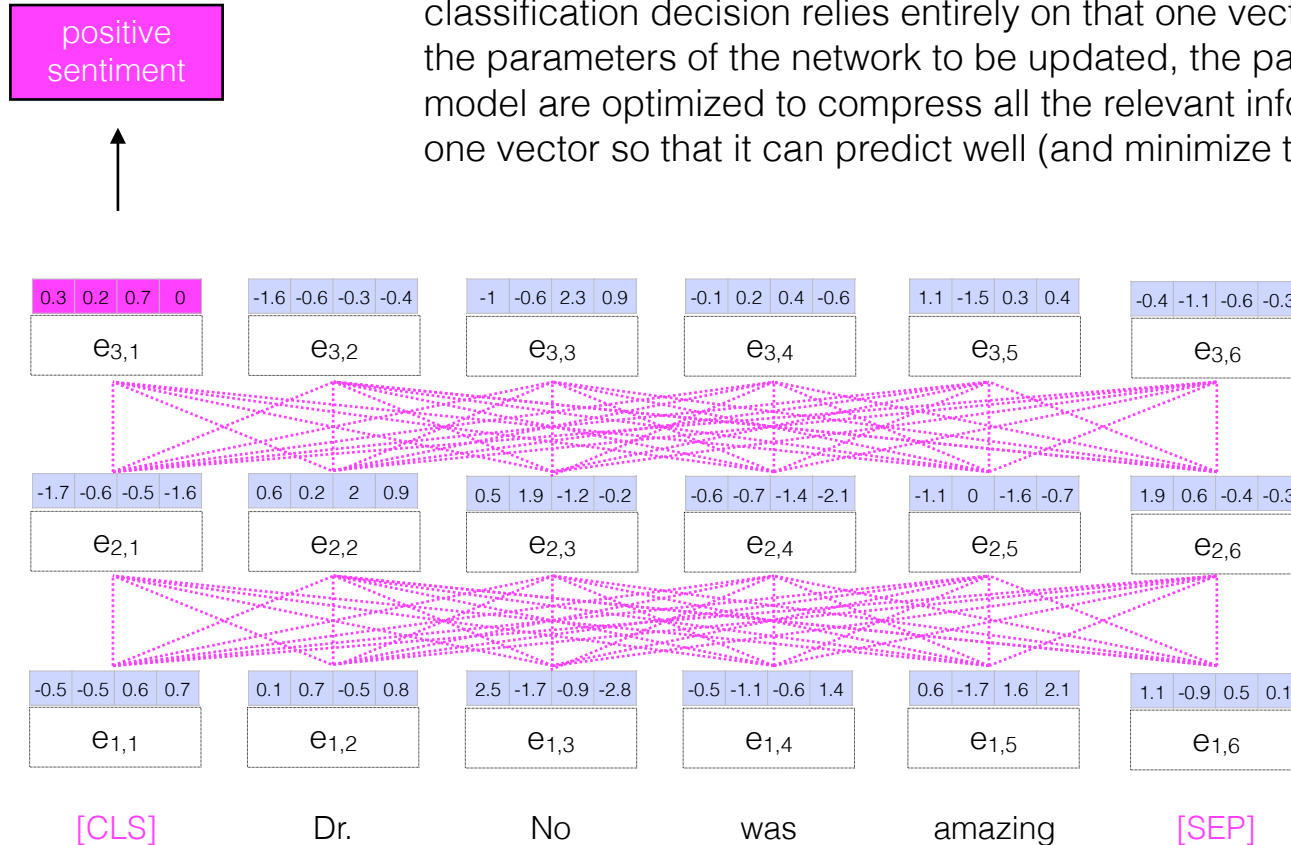


- These h separate output heads are then concatenated together and linearly transformed back to the original dimensionality d_v



why is pushing all the classification into a single token in BERT preferred is that not just making us lose information?

- We can represent the entire document with this *one* [CLS] vector
- Why does this work? When we design our network so that a classification decision relies entirely on that one vector and allow all the parameters of the network to be updated, the parameters of the model are optimized to compress all the relevant information into that one vector so that it can predict well (and minimize the loss).



Can we review CRF - specifically its similarity to Logistic Regression, its expression where we sum up certain features and how it is overcoming label bias

MEMM

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

For all training data, we want probability of the true label y_i conditioned on the previous true label y_{i-1} to be high.

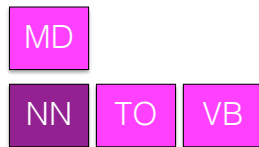
This is simply multiclass logistic regression

MEMM Training

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

- **Locally normalized** — at each time step, each conditional distribution sums to 1
- For a given conditioning context, the probability of a tag (e.g., **VBZ**) only competes against other tags with that same context (e.g., **NN**)

Label bias

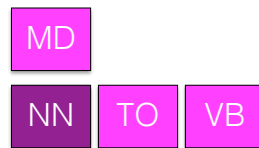


will to fight

	NN	MD
x_i =will	10	40
y_{i-1} =START	-1	7
BIAS	7	-2

Modals show up much more frequently at the start of the sentence than nouns do (e.g., questions); if we use greedy decoding, we'll select will=MD (though the problem still happens with Viterbi decoding)

Label bias



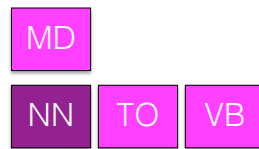
will to fight

MD + TO is *extremely* rare (“can to”) BUT *to* is relatively deterministic (almost always TO) **so it doesn't matter what tag precedes it.**

$$P(y_2=TO \mid y_1=NN, x, \beta) = 1.0$$

$$P(y_2=TO \mid y_1=MD, x, \beta) = 1.0$$

Label bias



will to fight

That means our prediction for *to* can't help us disambiguate *will*. We lose the information that MD + TO sequences rarely happen.

Conditional random fields

- We can solve this problem using global normalization (over the entire sequences) rather than locally normalized factors.
- This allows us to create features over the entire sequence and learn, for example, that **sequences** that have (“MD TO”) are bad but **sequences** with (“NN TO”) are good.

MEMM

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

CRF

$$P(y \mid x, \beta) = \frac{\exp(\Phi(x, y)^\top \beta)}{\sum_{y' \in \mathcal{Y}} \exp(\Phi(x, y')^\top \beta)}$$

Conditional random fields

$$P(y \mid x, \beta) = \frac{\exp(\Phi(x, y)^\top \beta)}{\sum_{y' \in \mathcal{Y}} \exp(\Phi(x, y')^\top \beta)}$$

Feature vector scoped over the entire input and label sequence

$$\Phi(x, y) = \sum_{i=1}^n \phi(x, i, y_i, y_{i-1})$$

ϕ is the same feature vector we used for local predictions using MEMMs

Features

$$\phi(x, i, y_i, y_{i-1})$$

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

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

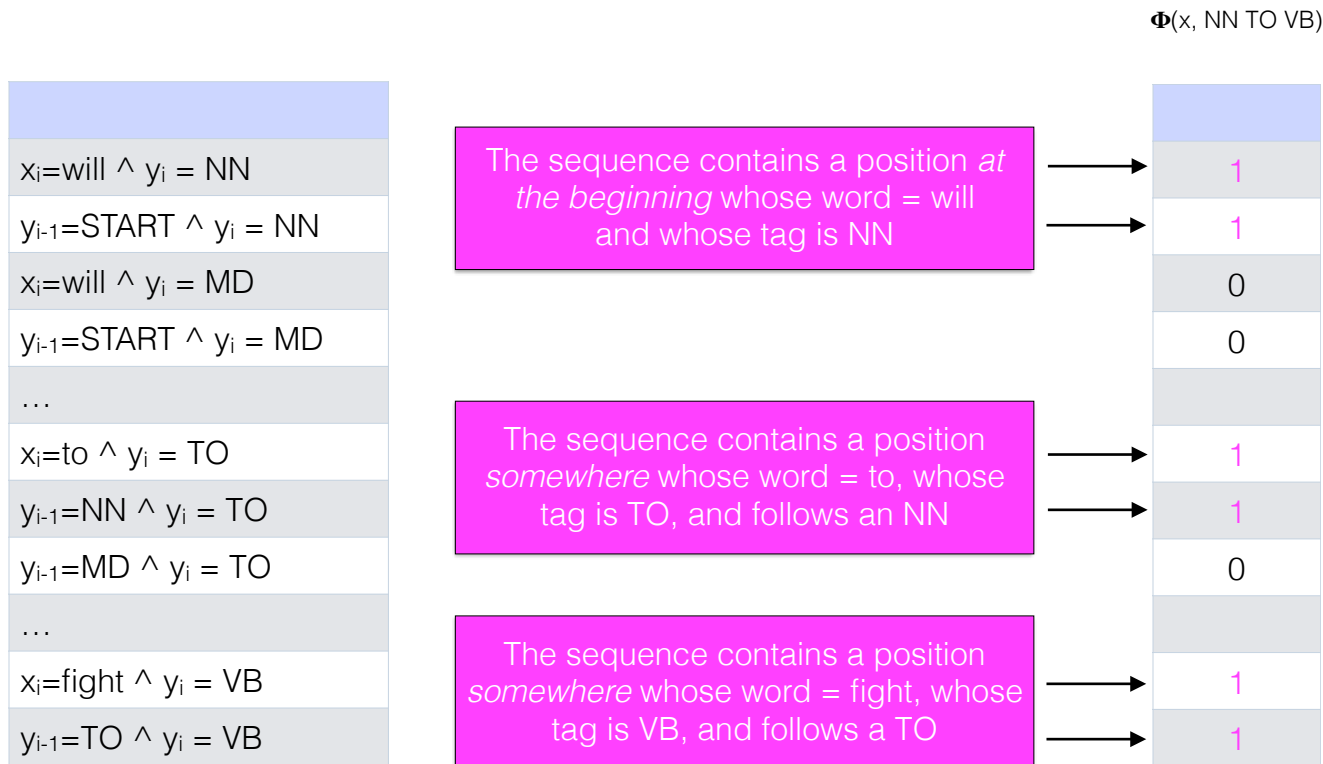
In an MEMM, we estimate $P(y_t \mid y_{t-1}, x, \beta)$ from each $\phi(x, t, y_t, y_{t-1})$ independently

	will $\phi(x, 1, y_1, y_0)$	to $\phi(x, 2, y_2, y_1)$	fight $\phi(x, 3, y_3, y_2)$
$x_i = \text{will} \wedge y_i = \text{NN}$	1	0	0
$y_{i-1} = \text{START} \wedge y_i = \text{NN}$	1	0	0
$x_i = \text{will} \wedge y_i = \text{MD}$	0	0	0
$y_{i-1} = \text{START} \wedge y_i = \text{MD}$	0	0	0
...			
$x_i = \text{to} \wedge y_i = \text{TO}$	0	1	0
$y_{i-1} = \text{NN} \wedge y_i = \text{TO}$	0	1	0
$y_{i-1} = \text{MD} \wedge y_i = \text{TO}$	0	0	0
...			
$x_i = \text{fight} \wedge y_i = \text{VB}$	0	0	1
$y_{i-1} = \text{TO} \wedge y_i = \text{VB}$	0	0	1

In a CRF, we use features from the entire sequence (by summing the individual features at each time step)

	will $\phi(x, 1, y_1, y_0)$	to $\phi(x, 2, y_2, y_1)$	fight $\phi(x, 3, y_3, y_2)$	$\Phi(x, \text{NN TO VB})$
$x_i = \text{will} \wedge y_i = \text{NN}$	1	0	0	1
$y_{i-1} = \text{START} \wedge y_i = \text{NN}$	1	0	0	1
$x_i = \text{will} \wedge y_i = \text{MD}$	0	0	0	0
$y_{i-1} = \text{START} \wedge y_i = \text{MD}$	0	0	0	0
...				
$x_i = \text{to} \wedge y_i = \text{TO}$	0	1	0	1
$y_{i-1} = \text{NN} \wedge y_i = \text{TO}$	0	1	0	1
$y_{i-1} = \text{MD} \wedge y_i = \text{TO}$	0	0	0	0
...				
$x_i = \text{fight} \wedge y_i = \text{VB}$	0	0	1	1
$y_{i-1} = \text{TO} \wedge y_i = \text{VB}$	0	0	1	1

In a CRF, we use features from the entire sequence (by summing the individual features at each time step)



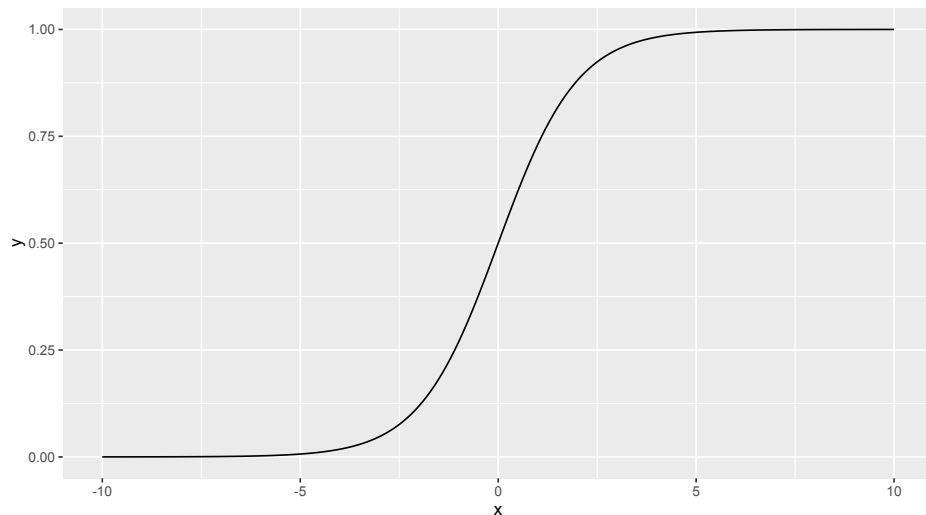
This lets us isolate the global sequence features that separate good sequences (in our training data) from bad sequences (not in our training data)

	$\Phi(x, \text{NN TO VB})$ GOOD	$\Phi(x, \text{MD TO VB})$ BAD	
$x_i = \text{will} \wedge y_i = \text{NN}$	1	0	these are the different (and so are potentially predictive of a good label sequence)
$y_{i-1} = \text{START} \wedge y_i = \text{NN}$	1	0	
$x_i = \text{will} \wedge y_i = \text{MD}$	0	1	
$y_{i-1} = \text{START} \wedge y_i = \text{MD}$	0	1	
...			
$y_{i-1} = \text{NN} \wedge y_i = \text{TO}$	1	0	these are the same (and so are not)
$y_{i-1} = \text{MD} \wedge y_i = \text{TO}$	0	1	
$x_i = \text{to} \wedge y_i = \text{TO}$	1	1	
$x_i = \text{fight} \wedge y_i = \text{VB}$	1	1	
$y_{i-1} = \text{TO} \wedge y_i = \text{VB}$	1	1	

Could we differentiate between the activations (Softmax, ReLU, Sigmoid, tanh) and regularization techniques (Lasso L1, Ridge L2, dropout, early stopping)?

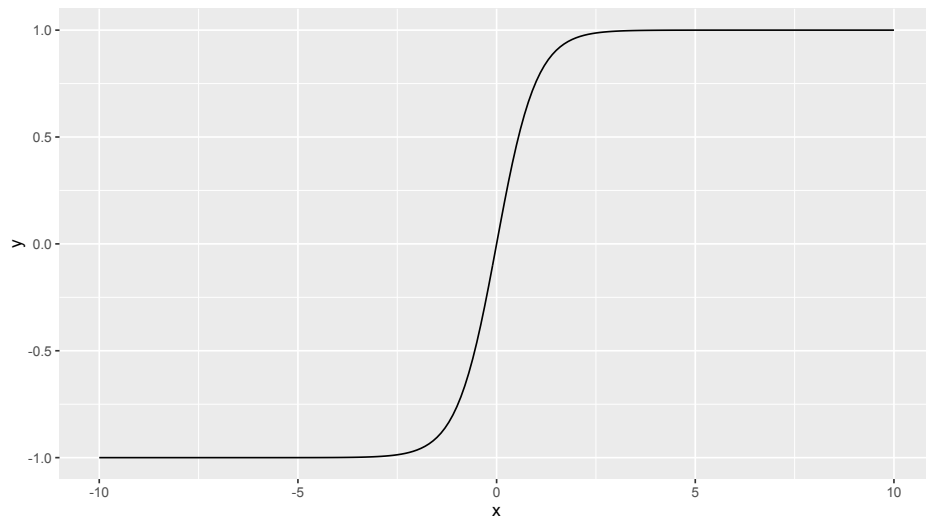
Activation functions

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$



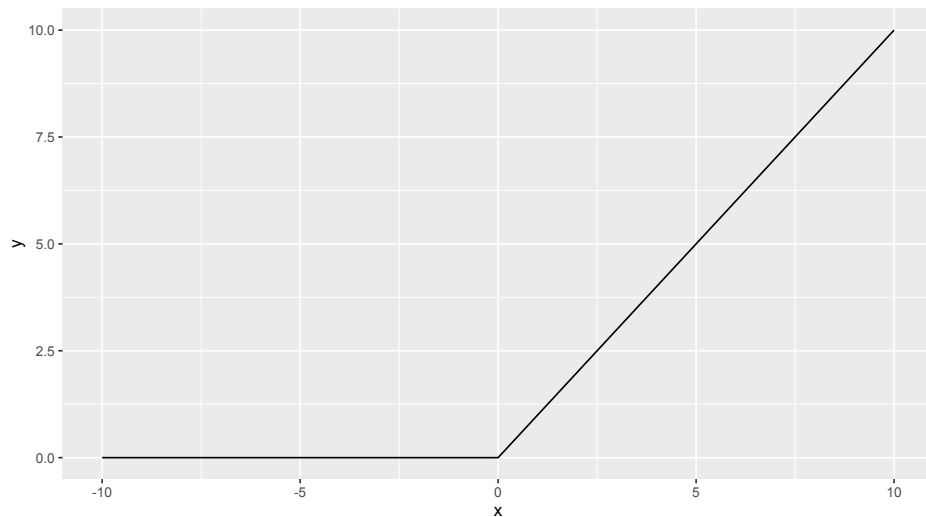
Activation functions

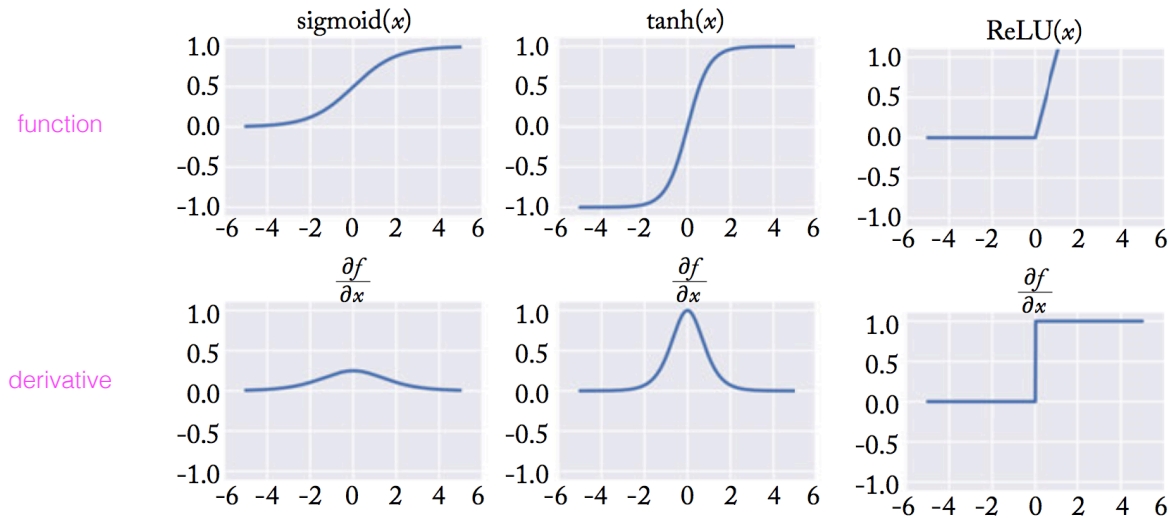
$$\tanh(z) = \frac{\exp(z) - \exp(-z)}{\exp(z) + \exp(-z)}$$



Activation functions

$$\text{ReLU}(z) = \max(0, z)$$





Goldberg 46

- ReLU and tanh are both used extensively in modern systems.
- Sigmoid is useful for final layer to scale output between 0 and 1, but is not often used in intermediate layers.

L2 regularization

$$\ell(\beta) = \underbrace{\sum_{i=1}^N \log P(y_i | x_i, \beta)}_{\text{we want this to be high}} - \underbrace{\eta \sum_{j=1}^F \beta_j^2}_{\text{but we want this to be small}}$$

- We can do this by changing the function we're trying to optimize by adding a penalty for having values of β that are high
- This is equivalent to saying that each β element is drawn from a Normal distribution centered on 0.
- η controls how much of a penalty to pay for coefficients that are far from 0 (optimize on development data)

L1 regularization

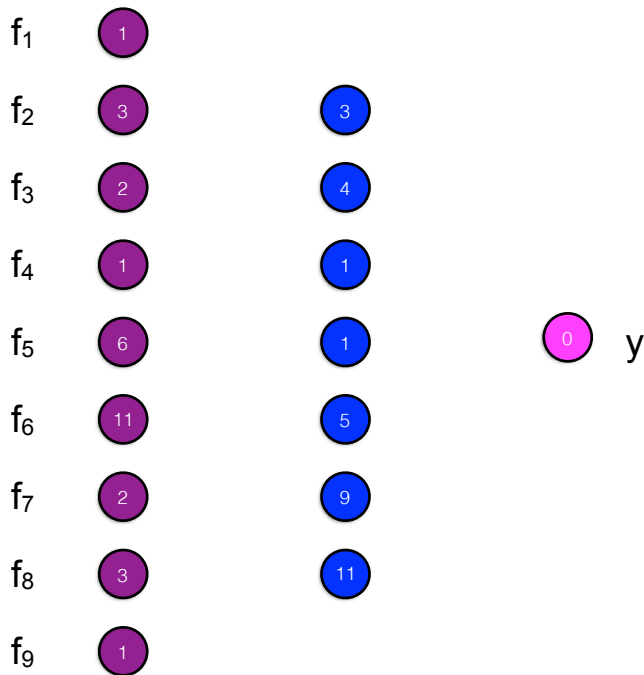
$$\ell(\beta) = \underbrace{\sum_{i=1}^N \log P(y_i \mid x_i, \beta)}_{\text{we want this to be high}} - \underbrace{\eta \sum_{j=1}^F |\beta_j|}_{\text{but we want this to be small}}$$

- L1 regularization encourages coefficients to be **exactly** 0.
- η again controls how much of a penalty to pay for coefficients that are far from 0 (optimize on development data)

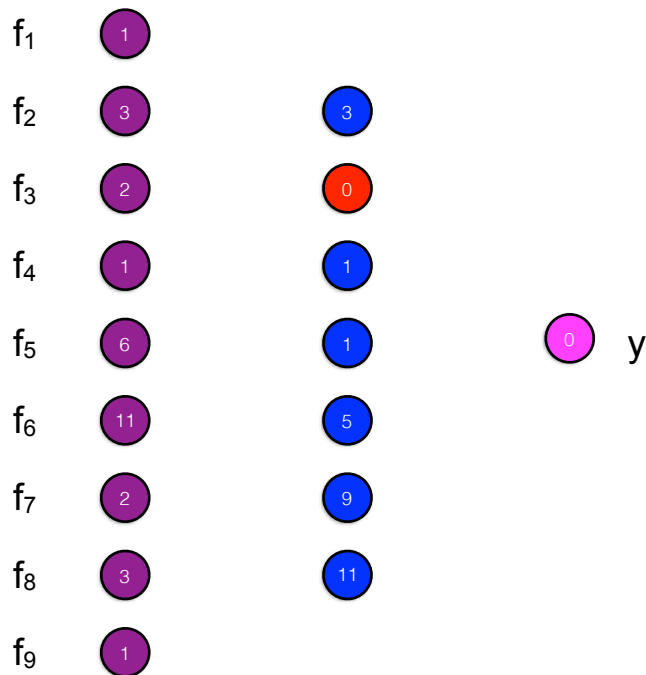
Regularization

- L1/L2 regularization: penalize W and V for being too large
- Dropout: when training on a $\langle x, y \rangle$ pair, randomly remove some node and weights.
- Early stopping: Stop backpropagation before the training error is too small.

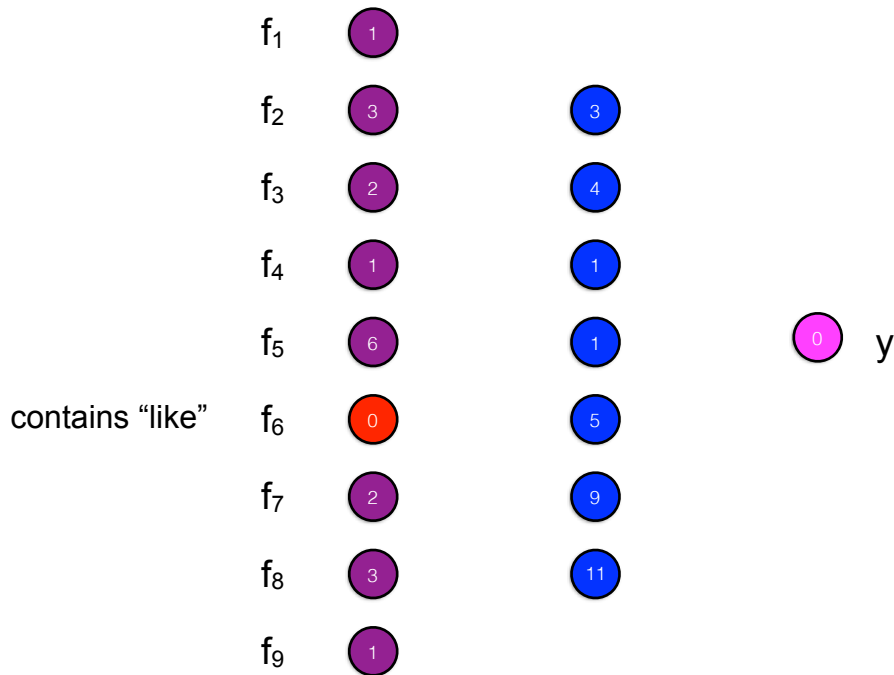
Dropout



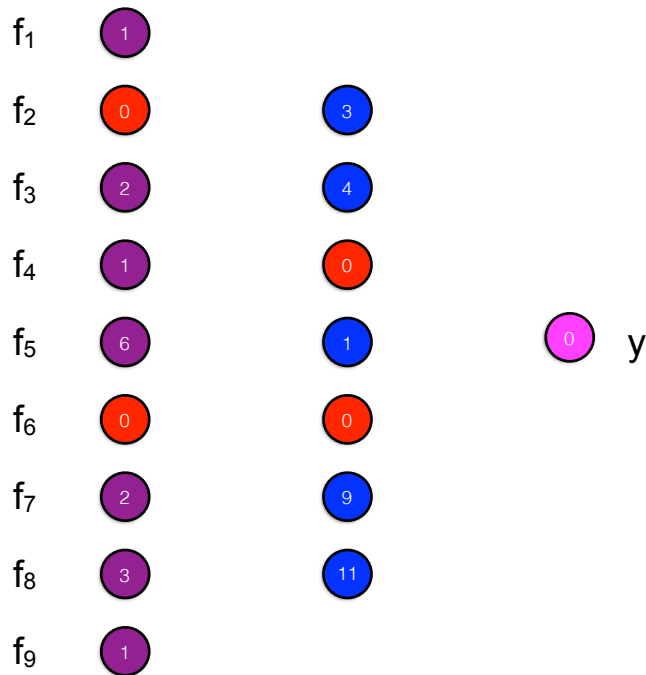
- Let's assume a FFNN (like a MLP) with a single hidden layer.
- Dropout removes nodes during training to encourage the model to not rely on them.
- Only active during train time, not at test time.



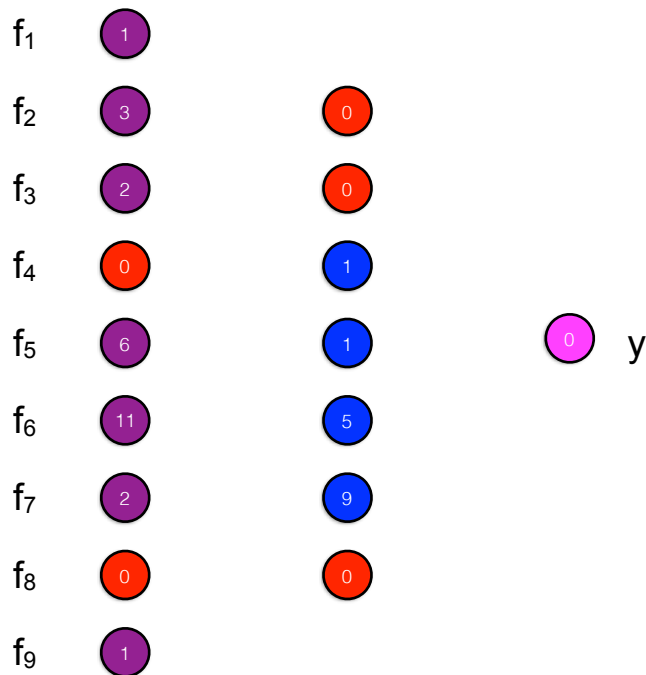
- Let's assume a FFNN (like a MLP) with a single hidden layer.
- Dropout removes nodes during training to encourage the model to not rely on them.
- Only active during train time, not at test time.



- Let's assume a FFNN (like a MLP) with a single hidden layer.
- Dropout removes nodes during training to encourage the model to not rely on them.
- Only active during train time, not at test time.



- Let's assume a FFNN (like a MLP) with a single hidden layer.
- Dropout removes nodes during training to encourage the model to not rely on them.
- Only active during train time, not at test time.



- Let's assume a FFNN (like a MLP) with a single hidden layer.
- Dropout removes nodes during training **to encourage the model to not rely on them**.
- Only active during train time, not at test time.