CS371N Lecture 10

LM2: Self-attention, Transformers

Announcements

- A2 due
- A3 out, due in 2 weeks
- Bias in embs response due

Recap Language models:

$$P(\bar{w}) = \prod_{i=1}^{\ell} P(w_i \mid w_1 \ldots w_{i-1})$$

n-gram LMs: $P(\bar{w}) = \prod_{i=1}^{\ell} P(w_i \mid w_{i-n+1} \ldots w_{i-1})$

Store probabilities explicitly (model as categorical distributions)

Estimation: count + normalize

Neural LMs:

predict $w_i \mid w_1 \ldots w_{i-1}$

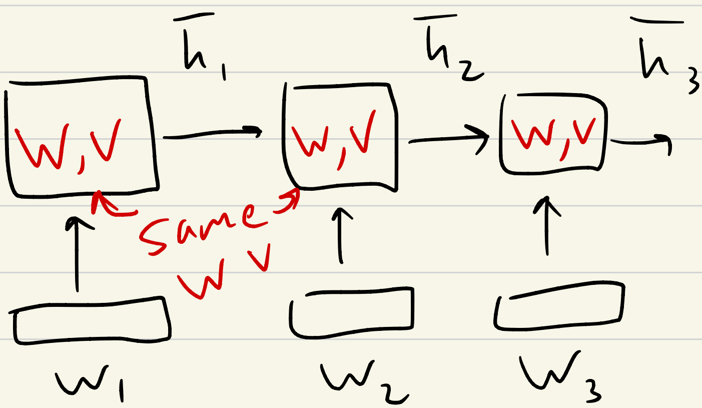
DANs? FFNNs?

<u>Today</u>

- RNNs and their shortcomings
- (Self)-attention

<u>RNNs</u> encode a sequence of vectors sequentially by tracking a "hidden state"

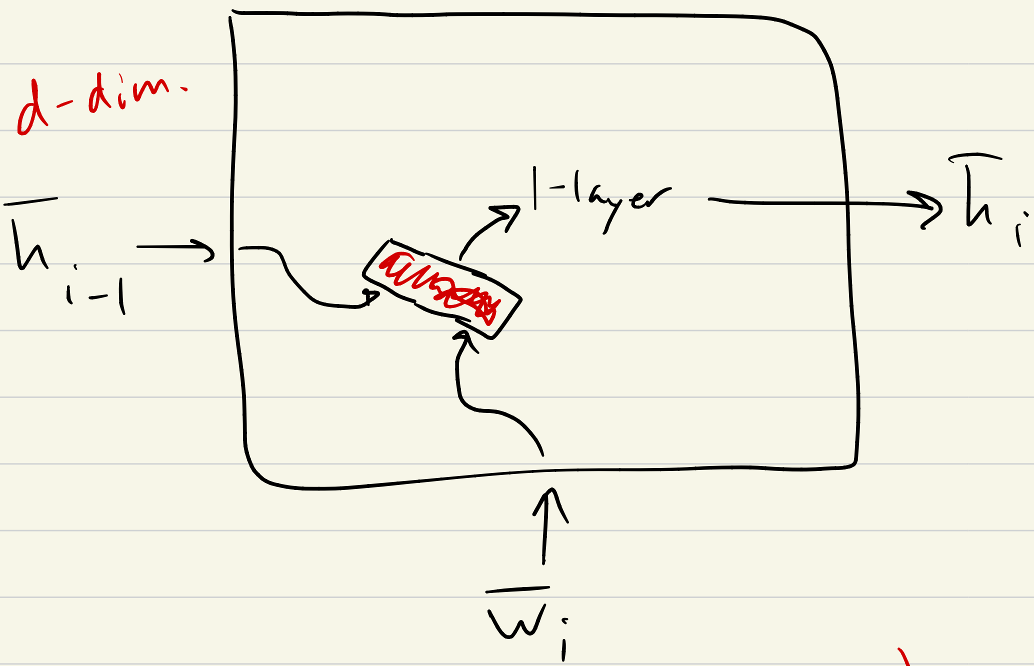
"summary" of the seq.



$$P(w_4 \mid w_1 \ldots w_3) = \text{softmax}(U\bar{h}_3)$$

Elman network

RNN cell



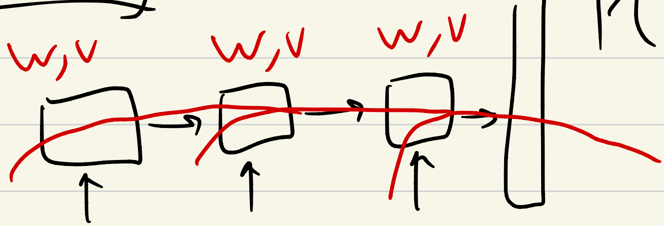
$$\bar{h}_i = \tanh(W\bar{w}_i + V\bar{h}_{i-1})$$

$$\left.\begin{array}{l} W: d \times |\text{word emb}| \text{ matrix} \\ V: d \times d \end{array}\right] \text{parameters}$$

Key property: params don't depend on sequence length
$\Rightarrow$ can scale to arbitrarily long inputs

However, it's position-sensitive!

<u>Training</u>



$P(w_4 \mid w_1\, w_2\, w_3)$
$\downarrow$
loss (NLL)

backprop

updates to W,V accumulate over whole sequence

Hard to learn! $\Rightarrow$ Long short-term memory net (LSTM)

Shortcomings of RNNs

"Forgetfulness": hard to track information over many steps

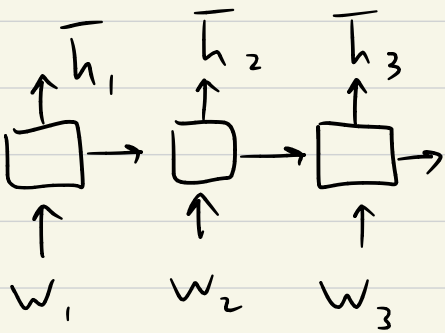
$$\bar{h}_i = \cancel{\tanh}(\sim + V\bar{h}_{i-1})$$

$$\bar{h}_i = V\bar{h}_{i-1} \qquad h_i \approx \sum_{j=1}^{i} V^{(i-j)} W w_j$$

LSTM: "gates" to control what parts of the vector change

$O(n)$ sequential dependence

RNN "API"

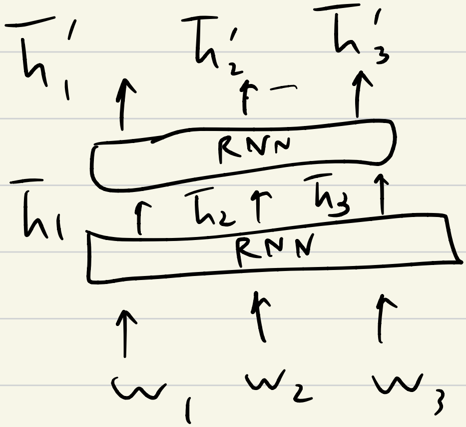


$\bar{h}_3$ "context-aware" word embedding for word w_3

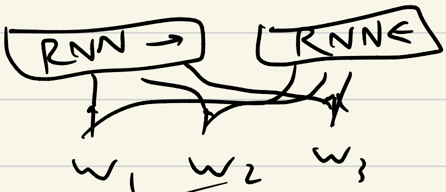
$\bar{h}_3$ blends $\bar{h}_2$ (context) w/ w_3 (word 3)

RNN(seq of vectors) $\Rightarrow$ seq of vectors aware of context

Stack these layers



Can increase depth



Transformer: layer that contextualizes words based on other words in the sequence

$$(e_1', e_2', e_3') = \text{Transformer}(e_1, e_2, e_3)$$

Running example:
Suppose we have seqs of As and Bs of length 4
if all As → next is A
if any B → next is B

AAAAA
ABAAB
BAABB

predict next char using this sequence that came before

$\underbrace{\text{BAAAA} \ldots \text{A}}_{1000}$ B

hard for RNNs to predict

Attention = allows us to do "random access" on the context to retrieve info we need

"Souped up" DAN, will add order information next time

k_i $\quad$ e_i

Keys: embedding of the sequence

query: vector representing what we want to find

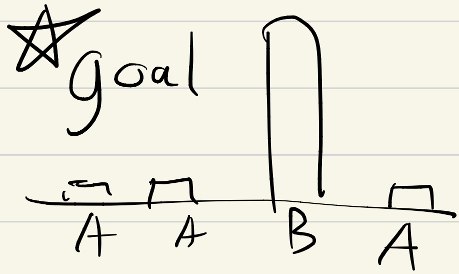
Assume key $A = \begin{bmatrix} 1 \\ 0 \end{bmatrix}^{e_A}$ $B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}^{e_B}$ (word embeddings)

$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$

A $\quad$ A $\quad$ B $\quad$ A

query: what we want to find

find Bs!



$q = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ "B"

Attention will compute a distribution over the tokens so far with higher weight for things that match q

<u>Steps</u> ① Compute score for each key based on query

$$S_i = K_i^T q$$

$$q = [0 \ 1]$$

$$
\begin{array}{lcccc}
S: & 0 & 0 & 1 & 0 \\
 & \begin{bmatrix} 1 \\ 0 \end{bmatrix} & \begin{bmatrix} 1 \\ 0 \end{bmatrix} & \begin{bmatrix} 0 \\ 1 \end{bmatrix} & \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\
 & A & A & B & A
\end{array}
$$

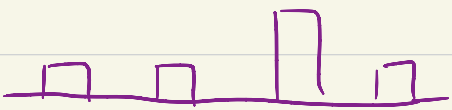
② Softmax scores to get probs.

$\bar{\alpha} = \text{softmax}(\bar{s})$

$[0 \quad 0 \quad 1 \quad 0]$ Assume $e=3$

$\downarrow$ $\qquad \dfrac{e^1}{e^0+e^0+e^1+e^0} \approx \dfrac{1}{2}$

$[1/6 \quad 1/6 \quad 1/2 \quad 1/6]$



③ Compute the output

$\text{output} = \sum \alpha_i e_i$ weighted sum of e_i

$= \frac{1}{6}\begin{bmatrix}1\\0\end{bmatrix} + \frac{1}{6}\begin{bmatrix}1\\0\end{bmatrix} + \frac{1}{2}\begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{6}\begin{bmatrix}1\\0\end{bmatrix}$

$= \begin{bmatrix}1/2\\1/2\end{bmatrix}$

Compare to DAN:

$$\frac{1}{4}\begin{bmatrix}1\\0\end{bmatrix} + \frac{1}{4}\begin{bmatrix}1\\0\end{bmatrix} + \frac{1}{4}\begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{4}\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}3/4\\1/4\end{bmatrix}$$

$\begin{bmatrix}1/2\\1/2\end{bmatrix}$ weights the B more highly

Ideally want:

$\begin{bmatrix}1\\0\end{bmatrix}$ if all As ✓

$\begin{bmatrix}0\\1\end{bmatrix}$ if any B $\begin{bmatrix}1/2\\1/2\end{bmatrix}$ ✗

Let $q = [0 \quad 10]$

$$\text{softmax}\,[0 \quad 0 \quad 10 \quad 0]$$

$$\downarrow$$

$0 \quad 0 \quad 1 \quad 0$ probs.

Decouple keys + queries from embeddings

Embedding matrix $E = \begin{array}{c} A \\ A \\ B \\ A \end{array}\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$

target e

$$\underbrace{\left(\overbrace{E^T W^K}^{K}\right)}\ \left(W^Q e\right) \qquad B$$

$$W^K = I \qquad W^Q = 10 I \qquad e = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow q = \begin{bmatrix} 0 \\ 10 \end{bmatrix}$$

(keep $\begin{bmatrix} 1 \\ 0 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix}$)

Parameters W^K and W^Q will let us learn how to query

Self-attention

every word is a key and query simultaneously

do one attention computation per word $\Rightarrow$ contextualized embeddings for each word

$(d=2)$

E: same embs, seq len $\times d$

K: same, seq len $\times d$

Q: seq len $\times d$ (rather than $1 \times d$)

Scores $S = Q K^T$

$\underset{\text{len} \times \text{len}}{S} = \underset{\text{len} \times d}{Q} \; \underset{d \times \text{len}}{K^T}$

$S_{ij} = q_i$ (ith row of Q)

$\cdot \, k_j$ (jth row of K)

$$\underbrace{4 \left[\quad \quad \quad \quad \right]}_{4}$$

all pairs of qs and ks

For now: $K = E$

$Q = E$

$$\begin{array}{rccc} & & \begin{array}{cccc} 1 & 1 & 0 & 1 \\ \downarrow & \downarrow & \downarrow & \downarrow \end{array} & S \\ \text{word 2} \rightarrow & \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} & \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} & = \begin{bmatrix} 1 & 1 & 0 & 1 \\ \multicolumn{4}{c}{\boxed{1 \quad 1 \quad 0 \quad 1}} \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \end{array}$$