CS 371N Lecture 10

LM 2: Self-attention, Transformers

Announcements

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

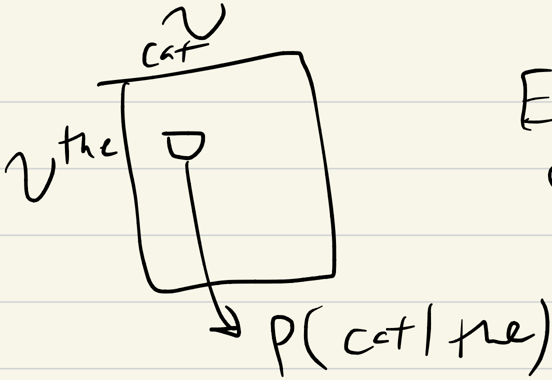
Recap Language models

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

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

Bigram: $P(w_i \mid w_{i-1})$

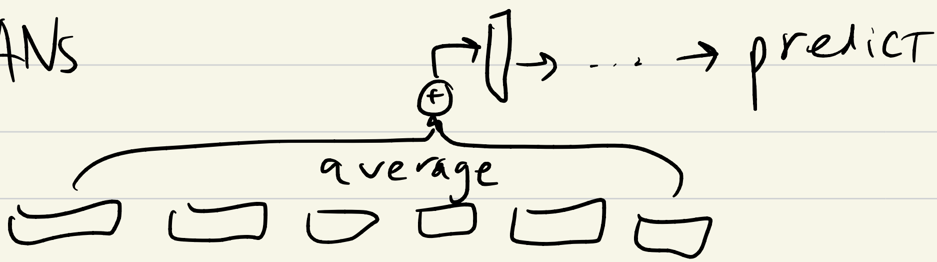
$\Rightarrow$ Explicitly model w/ categorical distribution



Estimate this by counting & normalizing

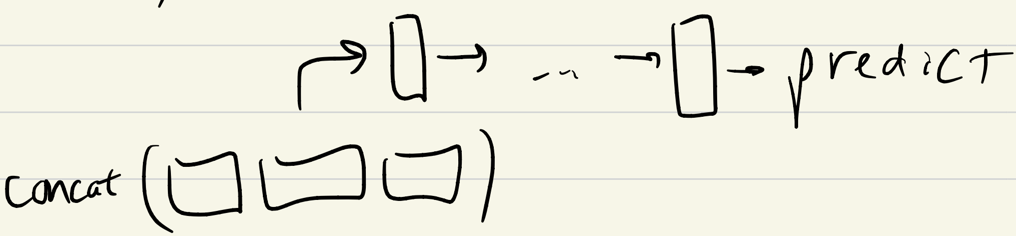
Neural LMs:

DANs



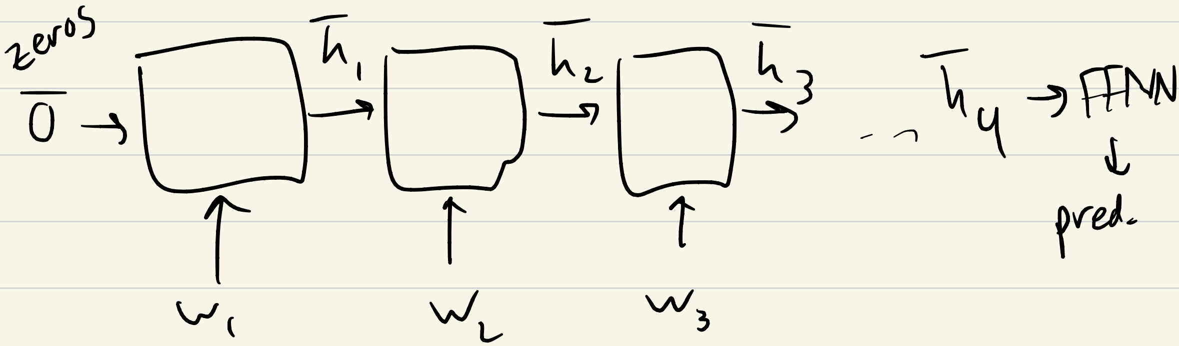
FFNNs that are "position sensitive"

Only consider n-1 words



<u>RNNs</u> Encode a sequence by repeatedly applying a "cell" to each input and passing a hidden state to the next cell

Predict $w_5 | w_1 \ldots w_4$



All $\square$ have the same params

What can this model do?

Example: add each w_i into $\bar{h}_{i-1}$

Example: only add w_i to $\bar{h}_{i-1}$ if it has a certain value

Why are RNNs good?

- Scale to long sequences
- "Complex enough" to fit hard tasks

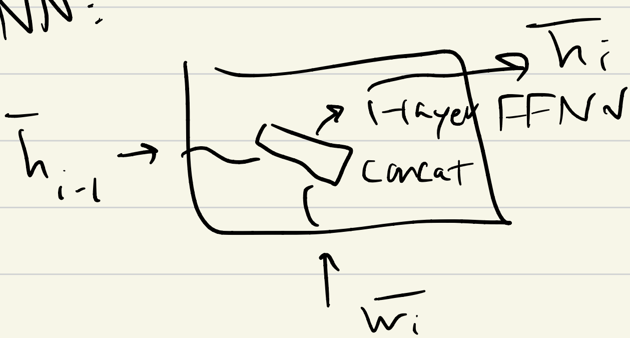
Why are RNNs bad?

- They "forget" over long strings

Imagine we're generating a story

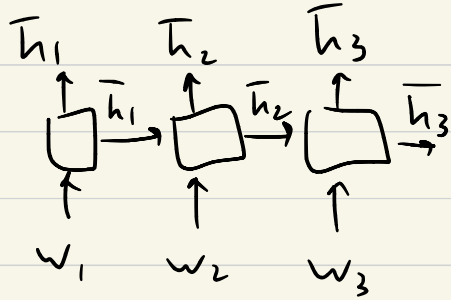
Sally looked around and saw a field.
It was nice . . . .
She went here. Someone said
"Hi ____"

Ex of RNN:



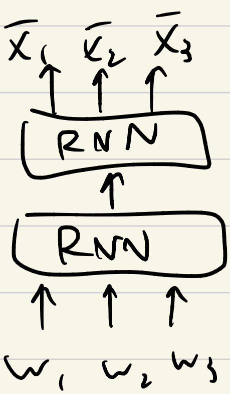
"API" for our neural nets

RNN:



$$\bar{h}_3 = \text{encode}(w_1, w_2, w_3)$$

$$\bar{h}_3 = \text{"context-sensitive encoding"}(w_3 \mid w_1, w_2)$$



Layers are stackable.

Transformer: obeys the same API.

Running example:

Suppose we have sequences of As and Bs of length 4

if all As → next is A

if any B → next is B

AAAAA   predict next char

BABBB   by scanning the sequence

BAAAB   for B

(a little like Sally)

Attention is a method of doing "random access" into the model's context to find info

Embeddings $e_1 \cdots e_4$ of the sequence

Keys $k_1 \cdots k_4$ (equals $e_1 \cdots e_4$ for now)

Query q representing what we want to find

Assume for A we have $e = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

B we have $e = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

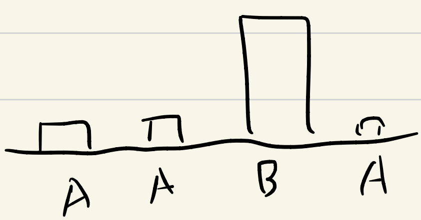
$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

A A B A

$q = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ because we want to find Bs

Attention computes a distribution over the keys given the query

Goal:



Steps ① Compute score for each key given query

$s_i = k_i^T q = [0\ 0\ 1\ 0]$

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

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

② Softmax scores to get probs

$\bar{\alpha} = \text{softmax}(\bar{s}) = [\tfrac{1}{6}\ \tfrac{1}{6}\ \tfrac{1}{2}\ \tfrac{1}{6}]$

Assume $e = 3$

$0 \rightarrow \dfrac{e^0}{e^0 + e^0 + e^1 + e^0} = \dfrac{1}{6} \qquad 1 \rightarrow \dfrac{e^1}{\text{-- --}} = \dfrac{1}{2}$

③ Compute output:

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

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

$= [1/2\ \ 1/2]$

Compare to DAN

$$\text{avg}\left(\begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix}\right) = \begin{bmatrix}3/4\\1/4\end{bmatrix}$$

so attention is "biased" towards B

What if we set

$$q = [0 \quad 10]$$

What new scores/α do we get?

scores $[0 \quad 0 \quad 10 \quad 0]$

$\alpha \approx [0 \quad 0 \quad 1 \quad 0]$

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

Decouple our keys + query from embeddings

Embedding matrix $E = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$

Matrices W^K and W^Q

"target" is B. To compute scores:

$$\underbrace{(E\, W^K)}\,\underbrace{(W^Q e)}$$

Suppose $W^K =$ identity $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Suppose $W^Q = 10 \cdot I$ $\begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$

This is equivalent to what we were doing before

$$\underset{4\times 2}{\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}}\ \underset{2\times 1}{\begin{bmatrix} 0 \\ 10 \end{bmatrix}} = \begin{bmatrix} 0 \\ 0 \\ 10 \\ 0 \end{bmatrix}$$

Self-attention

every word becomes a query executed against the keys

Sequence $e_1 \cdots e_n$

e_i is a query $\Rightarrow$ new value e_i' after attention

Map $e_1 \cdots e_n \to e_1' \cdots e_n'$ $\quad d=2$

E: seq len $\times d$ matrix $\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$

K: seq len $\times d$ $= EW^K$

Q: seq len $\times d$ $= EW^Q$

A A B A

Scores $S = QK^T$

seq len $\times$ seq len   seq len $\times d$ $d \times$ seq len

$S_{ij} = q_i$ (ith row of Q) $\cdot k_j$ (jth row of K)

$$S = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$ all pairs of ks and qs

Suppose $E = K = Q$

Version 2: let's use $W^Q = \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}$

Compute K, Q $W^K = \begin{bmatrix} 10 & \\ & 10 \end{bmatrix}$

Compute S

$$S = \begin{bmatrix} 10 & 10 & 0 & 10 \\ 10 & 10 & 0 & 10 \\ 0 & 0 & 10 & 0 \\ 10 & 10 & 0 & 10 \end{bmatrix}$$

$Q = E$

$K = 10E$

$10EE^T$

In reality: $W^Q \neq I$

W^Q is some other weights helping us find related stuff

This is quadratic