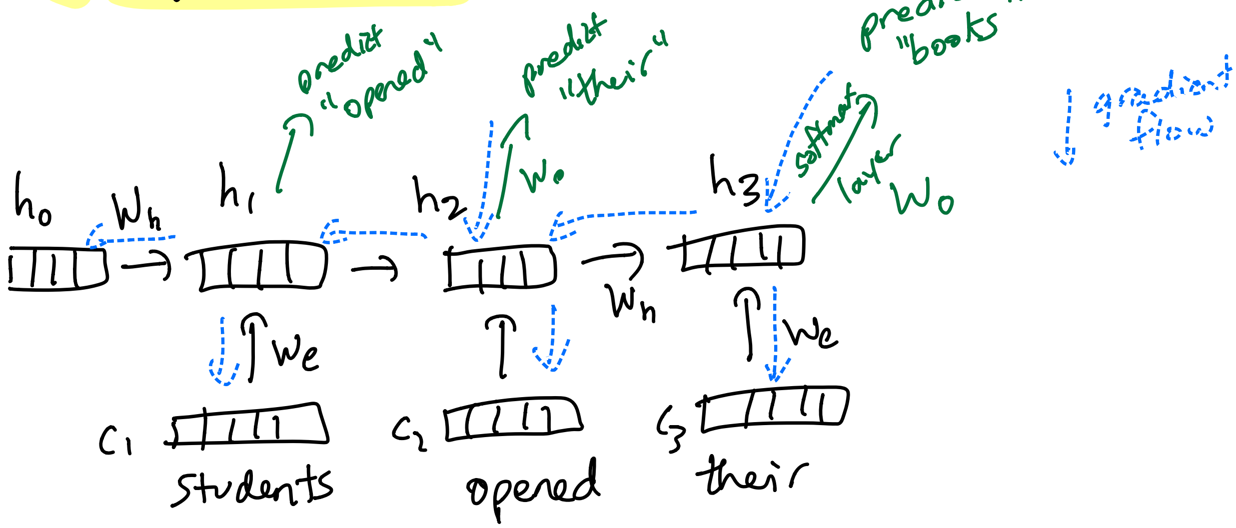
From RNNs to attention:

1. loss computation in RNN LMs
2. issues w/ RNN LM
   - ↳ bottleneck
   - ↳ cannot be parallelized across timesteps
3. self-attention

RNN LM loss



$$L_1 = -\log P(\text{opened} \mid \ldots)$$

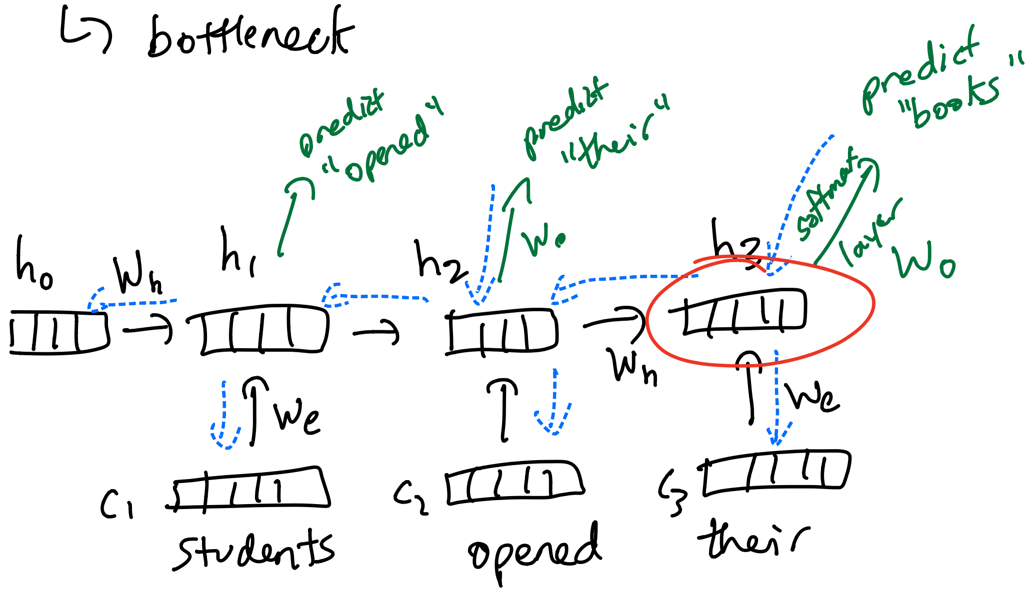
$$L_2 = -\log P(\text{their})$$

$$L_3 = -\log P(\text{books})$$

$$L = \frac{L_1 + L_2 + L_3}{3}$$ } avg. neg. log likelihood of the ground-truth next word over all tokens in the batch

Issues of RNNs:

↳ bottleneck



h_3 is required to encode <u>ALL</u> of the useful info from the prefix to predict the next word

↳ with long, complex prefixes, this is an unreasonable expectation

↳ <u>RAY MOONEY</u>: (~2014)

"you can't represent the meaning of a sentence in a BLEEPING vector"

attention mechanism

↳ a way to get around the bottleneck in neural LMs

↳ intuition: attention provides a way to access hidden states that are far away

↳ history: developed initially for RNNs ~2014, Bahdanau, Cho

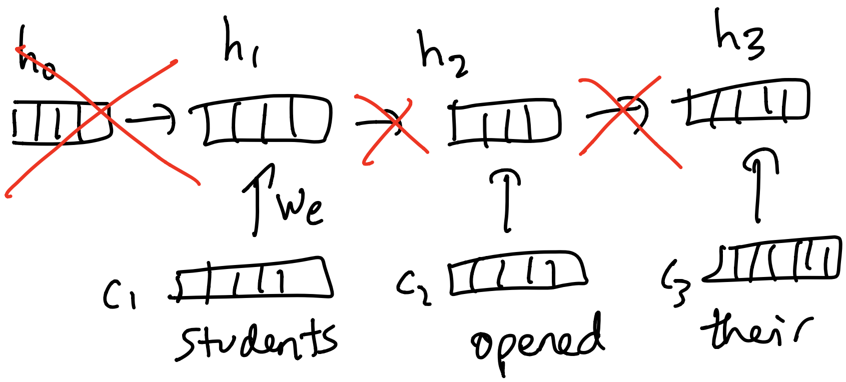
↳ now: "self-attention", core module of the Transformer LM

↳ introduced by Google in 2017

↳ reduces bottleneck effect

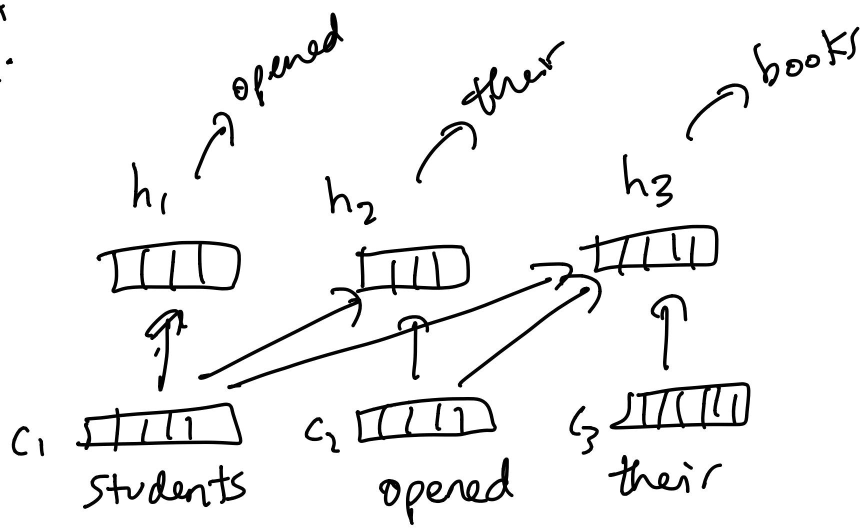
↳ fully parallelizable across timesteps

↳ hidden state at each timestep is independent of prev. hidden states



h_0
h_1
h_2
h_3
W_e
c_1
c_2
c_3
students
opened
their

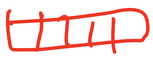
goal:



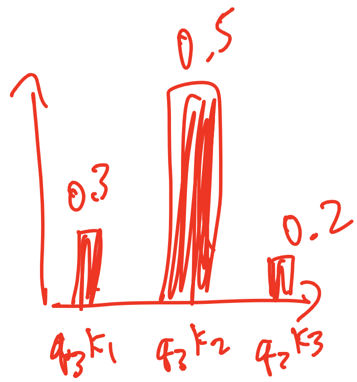
opened
their
books
h_1
h_2
h_3
c_1
c_2
c_3
students
opened
their

Computation of hidden state at timestep 3:

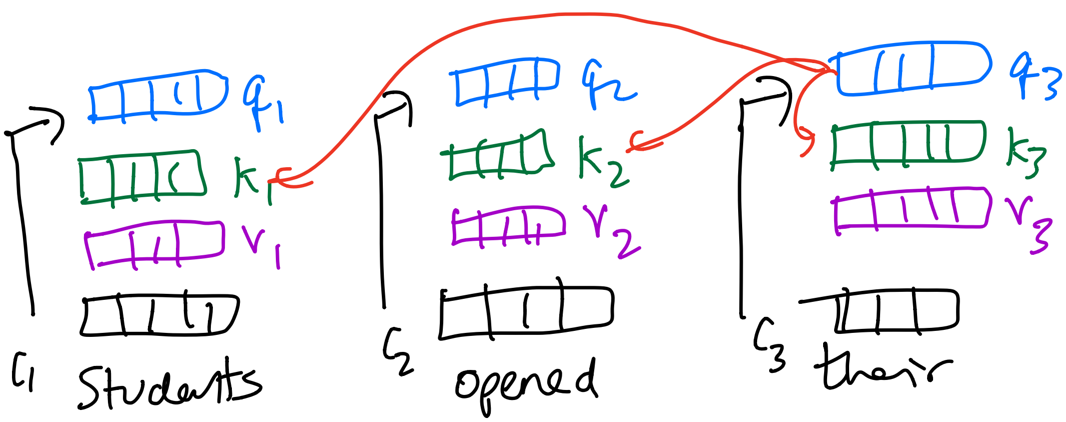
$h_3 = \underline{0.3}v_1 + \underline{0.5}v_2 + \underline{0.2}v_3$

= 

↳ softmax layer predict "books"



attn scores: $\text{softmax}(\langle q_3 \cdot k_1, q_3 \cdot k_2, q_3 \cdot k_3 \rangle)$



$q_1 = f(\underline{W_q}\, c_1) \Rightarrow$ query
$k_1 = f(W_k\, c_1) \Rightarrow$ key
} used to compute "attention score", use dot product $q \cdot k$

$v_1 = f(W_v\, c_1) \Rightarrow$ value } encode the information used to compute the hidden state

↑ weight matrices that are parameters of the model

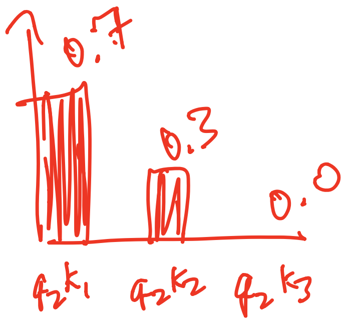
→ $q_2 = f(\underline{W_q}\, c_2)$

Second timestep:

$h_2 = 0.7v_1 + 0.3v_2$

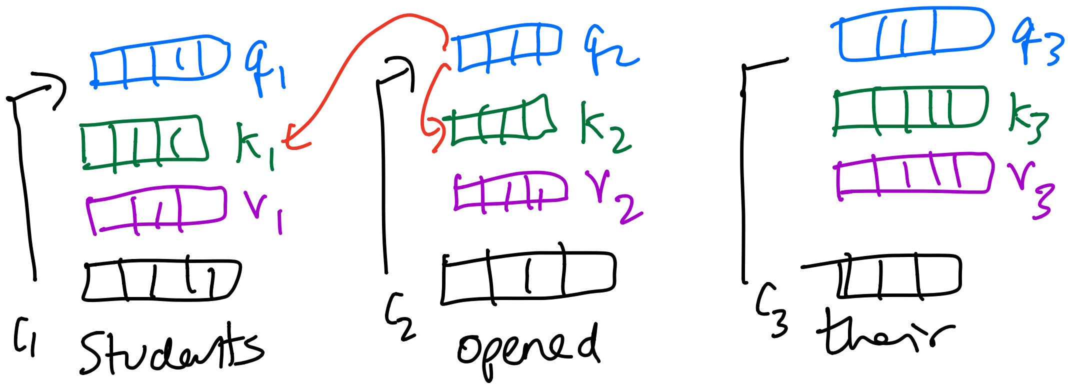
= [vector]

↳ predict "their"



no q_2k_3 bc of cheating!

attn scores: $\text{softmax}(\langle q_2k_1, q_2k_2 \rangle)$

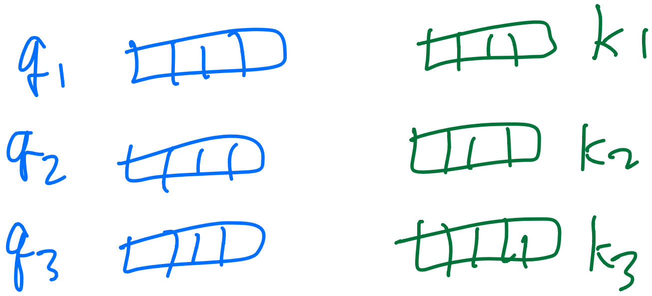


there are no dependencies between h_1, h_2, h_3!

↳ can parallelize

↳ reduced bottleneck

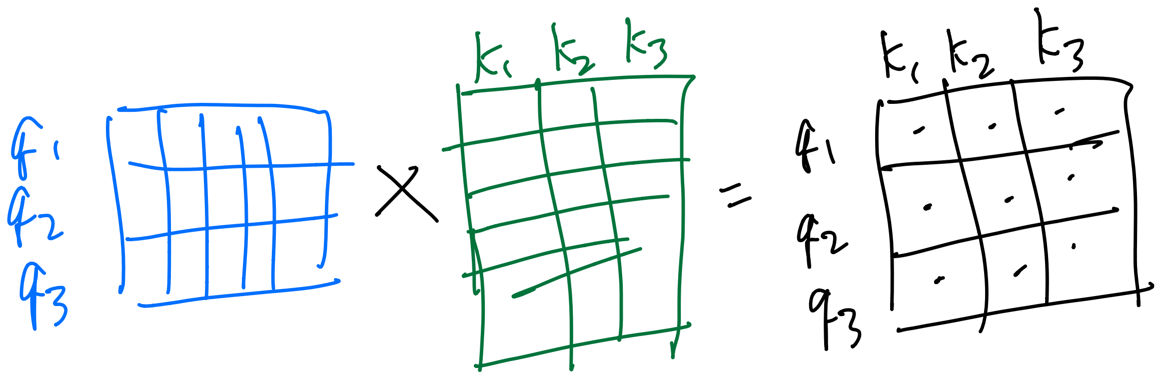
attn scores



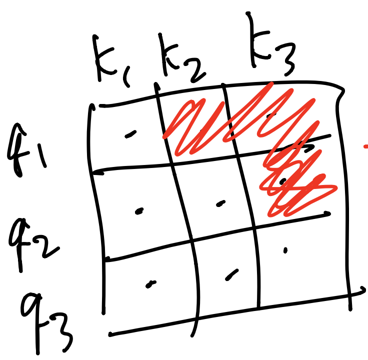
$a_1 = \langle q_1 k_1 \rangle$

$a_2 = \langle q_2 k_1, q_2 k_2 \rangle$

$a_3 = \langle q_3 k_1, q_3 k_2, q_3 k_3 \rangle$



one matrix product to compute all attn scores



these cells have info about the future. need to exclude

$$
\text{softmax}\left(
\begin{array}{c|ccc}
 & k_1 & k_2 & k_3 \\
\hline
q_1 & \cdot & \cdot & \cdot \\
q_2 & \cdot & \cdot & \cdot \\
q_3 & \cdot & \cdot & \cdot
\end{array}
\cdot
\begin{bmatrix}
1 & -\infty & -\infty \\
1 & 1 & -\infty \\
1 & 1 & 1
\end{bmatrix}
\right)
$$

attn scores

mask matrix

$$
\begin{array}{c|ccc}
 & k_1 & k_2 & k_3 \\
\hline
q_1 & 1 & 0 & 0 \\
q_2 & 0.3 & 0.7 & 0 \\
q_3 & 0.1 & 0.5 & 0.4
\end{array}
\times
\begin{array}{c|ccc}
v_1 & \; & \; & \; \\
v_2 & & & \\
v_3 & & &
\end{array}
=
$$

$$
\begin{array}{c|ccccc}
h_1 & \; & \; & \; & \; & \; \\
h_2 & & & & & \\
h_3 & & & & &
\end{array}
$$

we just computed all three hidden states w/ just a couple matrix products

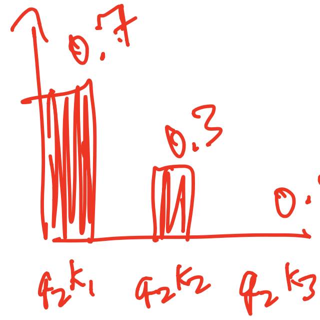
word order:

Second timestep:

$h_2 = 0.7 v_1 + 0.3 v_2$

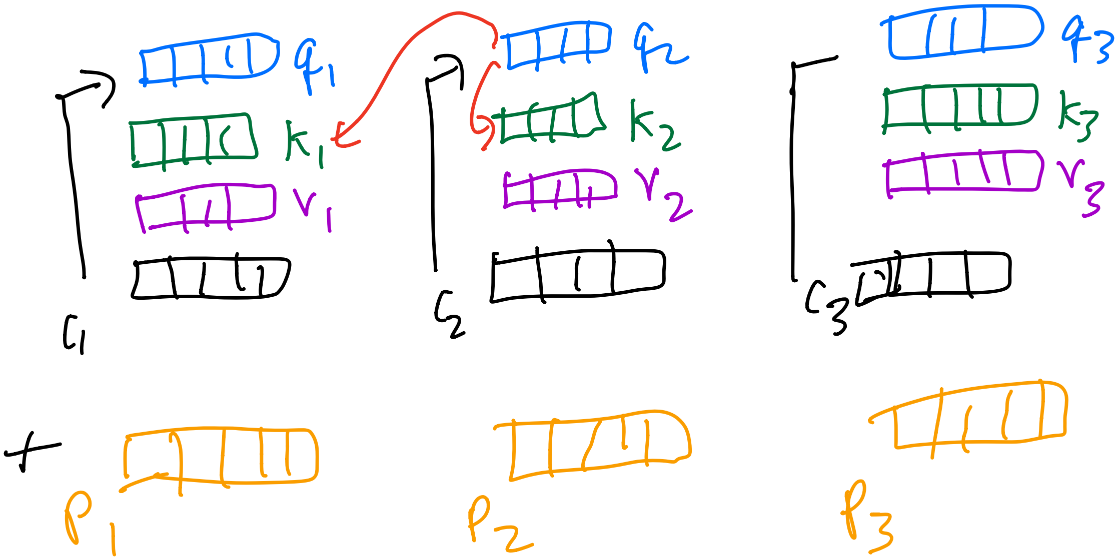
=

↳ predict "their"



→ no q_2k_3 bc of cheating!

attn scores: softmax($\langle q_2 k_1 , q_2 k_2$)



↳ positional embeddings ⇒ learned params

↳ p_1 is the same vector for the first pos. of every seq

↳ p_2 is the same vec for the second pos. of every seq

$emb = c_1 + p_1$