Position embeddings in Transformers:

- Without some explicit injection of position info, self-attention doesn't have any notion of order

Students opened their books

\[ \text{additive pos. embs} \]
\[ \text{absolute pos. embs} \]

absolute vs. relative pos embs

\[ \text{represent every pair of tokens in the input} \]

\[ q_{\text{students}} = W_q \cdot (c_{\text{students}} + p_1) \]

relative position embs:

- Generally cannot be added directly to input embs (RoPE is an exception)
- Instead directly modify the attn matrix
ALibi: $q_{\text{students}} = f(W_q \cdot C_{\text{students}})$

$k_{\text{books}} = f(W_k \cdot C_{\text{books}})$

$\Rightarrow$ intuitively, words that are closer together have a higher dot product

$\Rightarrow$ ALibi enables extrapolation beyond the training seq length

$\Rightarrow$ position info is only affecting $q, k$, but not $v$

Rotary position embs (RoPE)

- enables relative pos. embs without modifying the attn matrix like ALibi

- instead of adding pos. emb, we actually rotate the $q,k$ vectors via matrix/vector product w/ a rotation matrix
goal: dot product of rotated q, k
\( (q^T k) \) should be a function of
relative position only, not abs. pos.

ex.
\( c_1, c_2, c_3, c_4 \)
students opened their books

we want to compute \( q_4 \cdot k_1 \)

\( \text{RPE: find } f_q, f_k | g \) such that

\[
\begin{align*}
    f_q (c_{\text{books}}, 4) &= q_4 \\
    f_k (c_{\text{students}}, 1) &= k_1
\end{align*}
\]

\( q_4 \cdot k_1 = g (c_{\text{books}}, c_{\text{students}}, 3) \)
\( \Leftrightarrow q_4, k_1 \)

\( \implies \) this can be accomplished by
rotating \( W_q \) \& \( W_k \) by diff. angles

\[
\begin{align*}
    f_q (c_{\text{books}}, 4) &= R_{\theta_q, 4} \cdot W_q c_{\text{books}} \\
    f_k (c_{\text{students}}, 1) &= R_{\theta_k, 1} \cdot W_k c_{\text{students}} \\
    q &= q^T k
\end{align*}
\]
\[ R_{\theta, it} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \]

where \( \theta \) is hyperparameter