# Lec-4

## Introduction to Self Attention with trainable weights

(aka scaled dot product attention)

* Raw dot product b/t embeddings can give incorrect or ambiguous attention weights.

Eg,

The dog chased the ball but couldn't catch it.

→ "it" should attend more to ball.

→ raw dot product may give equal scores.

→ model cannot fix this without training.

* Instead of directly comparing input embedding, we do transform first and then compute attention in a learned space. This introduces trainable parameters.

Transforming embedding to Q, K, V space,

| I/P Embedding | Transforming matrix | Transformed embeddings |
|---|---|---|
| $X \rightarrow (5 \times 3)$ | $W_q \rightarrow (3 \times 2)$ | $XW_q \rightarrow Q \rightarrow 5 \times 2$ |
| | $W_k \rightarrow (3 \times 2)$ | $XW_k \rightarrow K \rightarrow 5 \times 2$ |
| | $W_v \rightarrow (3 \times 2)$ | $XW_v \rightarrow V \rightarrow 5 \times 2$ |

• Assuming we have 5 i/p tokens of 3 dim

* The dim of transforming weight matrices need not to be sam dim as the i/p embedding. In this case it is 2 dim.

* Therefore, the resulting transformed embeddings is of 2 dim.

* Attention score is calculated by taking dot product b/t transformed embeddings.

$$\boxed{Score = Q \cdot K^T}$$

* The scores are then scaled down to a factor of $1/\sqrt{d_k}$, then softmax probability is calculated.

$$\boxed{Weights = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)}$$

* Reason for scaling,

↳ Stability in training. Otherwise softmax values can become large [sharp softmax distribution]

↳ To make variance of dot product of $QK^T$ stable.

* Context vector is computed by taking weighted attention sum of value vector.

$$\boxed{Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V}$$

* This is what appears in "Attention is all you need" paper.

* At the end, each token will have one context vector.

——x——