LLM from Scratch - Elias Hossain

(1) Self-Attention Mechanism

Self-attention mechanism that allows a model to weigh different part of input sequence when computing representation for each token.

Given an input sequence X of length n with each token represented as an embedding vector x_i, self attention computes a weighted sum of all token embeddings for each token.

→ Query (Q): Represents the current token

→ key (K): Represents each token's importance in the context.

→ Value (V): Represents the token's actual contribution to the output.

The attention scores are computed using the scaled-dot product attention:

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

Where,

- Q, K, V are matrices formed from the input embeddings after linear transformation.
- d_k is the dimension of the key vectors.
- Softmax normalizes the scores to sum to 1.

P.T.O

Mathmeatiecal steps:

1/ Compute Queries, keys and Vailes:

$Q = XW_Q$, $K = XW_K$, $V = XW_V$

Where, W_Q, W_K, W_V are learnable weight matrices.

2/ Compute attentiom Scores:

$$A = \frac{QK^T}{\sqrt{d_k}}$$

3/ Apply Softmax:

$$\alpha = \text{Softmax}(A)$$

4/ Compute output:

$$Z = \alpha V$$

Causal (masked) Attention

Causal attention is a type of self-attention where tokens cannot attend to future tokens. This is essential for autoregressive models like GPT, ensuring that predictions only depend on past information.

Theory — This attention is implemented using a masking matrix, which sets future token attention scores to $-\infty$ before applying the softmax activation.

Mathmatical Implementation:

① Create a lower triangular mask M:

$$M_{ij} = \begin{cases} 0 & \text{if } j \le i \\ \infty & \text{if } j > i \end{cases}$$

② Modify attention scores:

$$A_{masked} = \frac{QK^T}{\sqrt{d_k}} + M$$

③ Apply softmax and compute the final weighted sum:

$$\alpha = \text{Softmax}(A_{masked})$$

$$Z = \alpha V$$

This ensures that each token can only attend to itself and previous token.

P.T.O

Example of a masking matrix:

For a sequence length of $n=5$, the masking matrix M looks like this:

$$M = \begin{bmatrix} 0 & -\infty & -\infty & -\infty & -\infty \\ 0 & 0 & -\infty & -\infty & -\infty \\ 0 & 0 & 0 & -\infty & -\infty \\ 0 & 0 & 0 & 0 & -\infty \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- This matrix is added to the attention score matrix before applying Softmax. It ensures that future tokens have a weight of nearly zero.

BOOM

Multi-Head Attention:

Multi-head attention (MHA) extends self-attention by applying multiple attention mechanisms in parallel, allowing the model to capture different types of relationships.

Theory

Instead of computing a single set of Q, K, V, MHA splits the input into multiple attention heads, each with its own learned weight matrices. The outputs are then concatenated and projected back into the original embedding space.

Mathematical Formulation:

(1) Compute seperate projections:

$$Q_i = XW_Q^i, \quad K_i = XW_K^i, \quad V_i = XW_V^i$$

(2) Compute attention output:

$$Z_i = \text{Softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_k}}\right) V_i$$

(3) Merged all heads:

$$Z = \text{Concat}(Z_1, Z_2, \cdots Z_n)$$

(4) Apply final linear transformation:

$$Z_{final} = ZW_O$$

Where W_O is a learnable weight matrix for projection back to the model's embedding space.