EECS 182 Deep Neural Networks
Fall 2026 Gireeja Ranade

Discussion 11

1. Query-Key-Value Mechanics in Self-Attention

Self-attention is a core building block of transformers. In this problem, we study query–key–value interactions among three hidden states from an RNN encoder. The hidden states and projection matrices are:

$$h_1 = \begin{bmatrix} 5 \\ 1 \end{bmatrix} \quad h_2 = \begin{bmatrix} 4 \\ 3 \end{bmatrix} \quad h_3 = \begin{bmatrix} -2 \\ 6 \end{bmatrix} \quad W_q = \begin{bmatrix} 1 & 0 \\ 2 & 9 \end{bmatrix} \quad W_k = \begin{bmatrix} 0 & 1 \\ 6 & 0 \end{bmatrix} \quad W_v = \begin{bmatrix} 4 & 3 \\ 9 & 1 \end{bmatrix}$$

Here, h_i is the hidden state at time step i. Each q_i, k_i, v_i is determined by $q_i = W_q h_i$, $k_i = W_k h_i$, $v_i = W_v h_i$.

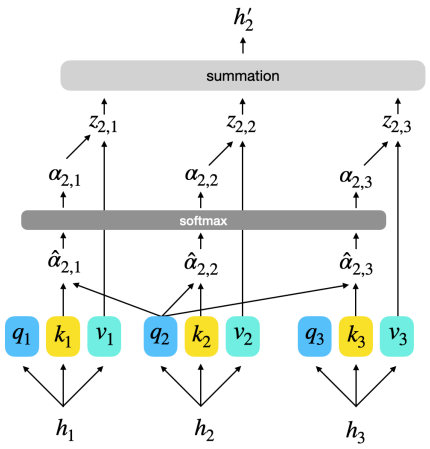


Figure 1: Self-attention at the second sequence position. Each $z_{2,j} = \alpha_{2,j} v_j$; in particular, $z_{2,1} = \alpha_{2,1} v_1$.

Use unscaled dot-product scores $\hat{\alpha}_{i,j} = q_i^T k_j$ in this exercise, omitting the usual $1/\sqrt{d_k}$ factor.

(a) **Compute $\hat{\alpha}_{2,1}$, $\hat{\alpha}_{2,2}$, $\hat{\alpha}_{2,3}$.**

(b) Figure 1 shows self-attention at the second sequence position. **Write equations for $h'_2 = \text{SelfAttention}(h_1, h_2, h_3)$.** You may define intermediate variables.

 All Rights Reserved. This may not be publicly shared without explicit permission.

(c) For simplicity, replace softmax with **hard attention**, an operation that takes the three scores $(\hat{\alpha}_{2,1}, \hat{\alpha}_{2,2}, \hat{\alpha}_{2,3})$ and outputs a one-hot vector: 1 at the position of the largest input and 0 at the other two positions. What is h'_2 in this case?

(d) In practice, it is common to have multiple self-attention operations happening in one layer. Each self-attention block is referred to as a *head*, and thus the entire block is typically called *Multi-head Self-Attention (MSA)* in papers. **What's the benefit of having multiple heads?** (*Hint*: why do we want multiple kernel filters in ConvNets?)

 All Rights Reserved. This may not be publicly shared without explicit permission.

2. NoPE (No Positional Encoding)

In models with a decoder-only architecture (such as GPT) using causal attention, it might be possible for the attention layer itself to learn to encode positional information.

Suppose we have a sequence of tokens $\{< beg >, x_2, \ldots, x_n\}$ where $< beg >$ is a special token indicating the beginning of the sequence. Additionally, suppose we learn an embedding where $< beg >$ is represented by d-dimensional vector $[1, 1, e_{b,3}, \ldots, e_{b,d}]$ and each other token x_i is represented by $[1, 0, e_{i,3}, \ldots, e_{i,d}]$ for $i = 2, 3, \ldots, n$. Here, $e_{b,j}$ and $e_{i,j}$ can be arbitrary when $j \geq 3$.

Additionally, suppose we have an attention head with query, key, and value matrices $W^Q \in \mathbb{R}^{d\times d}$, $W^K \in \mathbb{R}^{d\times d}$, $W^V \in \mathbb{R}^{d\times d}$ with the following learned values:

$$W^K = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \quad W^V = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

and W^Q is arbitrary.

Use row-vector token embeddings, with weight matrices multiplied on the right.

(a) **Write the embedding matrix $X \in \mathbb{R}^{n\times d}$ for the token sequence.**

(b) **Compute the key, query, and value matrices $K = XW^K$, $Q = XW^Q$, and $V = XW^V$.**

(c) **Calculate the masked scores $\alpha = \mathrm{Mask}(QK^T)$ and the attention weights after applying softmax to each row.**

(d) **Apply attention to V and compute the output for token x_t.**

 All Rights Reserved. This may not be publicly shared without explicit permission.

(e) **Explain why this output encodes the position t and how it can pass positional information to subsequent layers.**

Reference: https://arxiv.org/abs/2305.19466.

 © Faculty teaching EECS 182, Fall 2026. All Rights Reserved. This may not be publicly shared without explicit permission.

3. Transformer Architecture

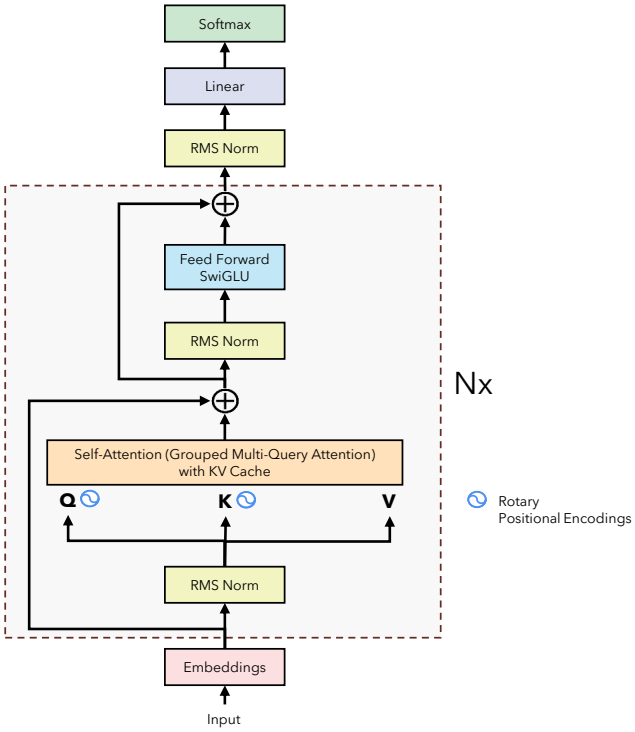


Figure 2: LLaMA architecture diagram. Many modern transformers use rotary positional embeddings (RoPE) and the SwiGLU activation function (source: https://github.com/hkproj/pytorch-llama-notes/).

(a) **What is the advantage of multi-head attention?** Give some examples of linguistic structures that can be found using multi-head attention (e.g. subject-verb agreement, translation between languages).

(b) Consider hard attention with one-hot argmax choices and residual connections. Fix the argmax choices. **After n layers, what is the maximum number of input positions contributing to a node through the selected value edges and residual paths?** Let the sequence length be L.
With a single head? With two heads? With h heads?

 All Rights Reserved. This may not be publicly shared without explicit permission.

(c) For input sequences of length M and output sequences of length N, **what are the complexities of (1) Non-Causal Self-Attention[1] (on the input sequence), (2) Non-Causal Self-Attention (on the input sequence), but with multi-query attention, (3) Cross Attention (from the output sequence to the input sequence), (4) Causal Self-Attention (on the output sequence).** Let d be the hidden dimension of the network (both the encoder and the decoder part), and h be the number of heads.

Recall: Normal multi-head attention is:

$$\text{MultiHeadedAttention}(X, X, X) = \text{Concat}_{i \in [\#\text{ heads}]} \left(\text{Attention} \left(XW_i^Q, XW_i^K, XW_i^V \right) \right) W^O$$

For this question, we assume W_i^Q, W_i^K, W_i^V are of shape $d \times d$, $W^O \in \mathbb{R}^{hd \times d}$, and X has shape $M \times d$ where each row of X is a token embedding. For cross-attention, the query sequence has N rows and the key/value sequence has M rows. We also assume that we use naive matrix multiplication, that is, if A, B are of shape $m \times n, n \times k$, then AB takes mnk multiply-accumulate operations.

With multi-query attention, there is just one W^K, W^V, thus:

$$\text{MultiQueryAttention}(X, X, X) = \text{Concat}_{i \in [\#\text{ heads}]} \left(\text{Attention} \left(XW_i^Q, XW^K, XW^V \right) \right) W^O$$

(d) **How many operations does the multi-query attention save compared to multi-head attention? What are some other advantages?**

Contributors: Matteo Guarrera, Mert Cemri, Sizhe Chen, Gireeja Ranade, Kevin Li, Lance Mathias, Anant Sahai, Olivia Watkins, Jerome Quenum, Saagar Sanghavi, Yuxi Liu, Qiyang Li.

[1] This is the "normal" self-attention from lecture.

 All Rights Reserved. This may not be publicly shared without explicit permission.