8.5

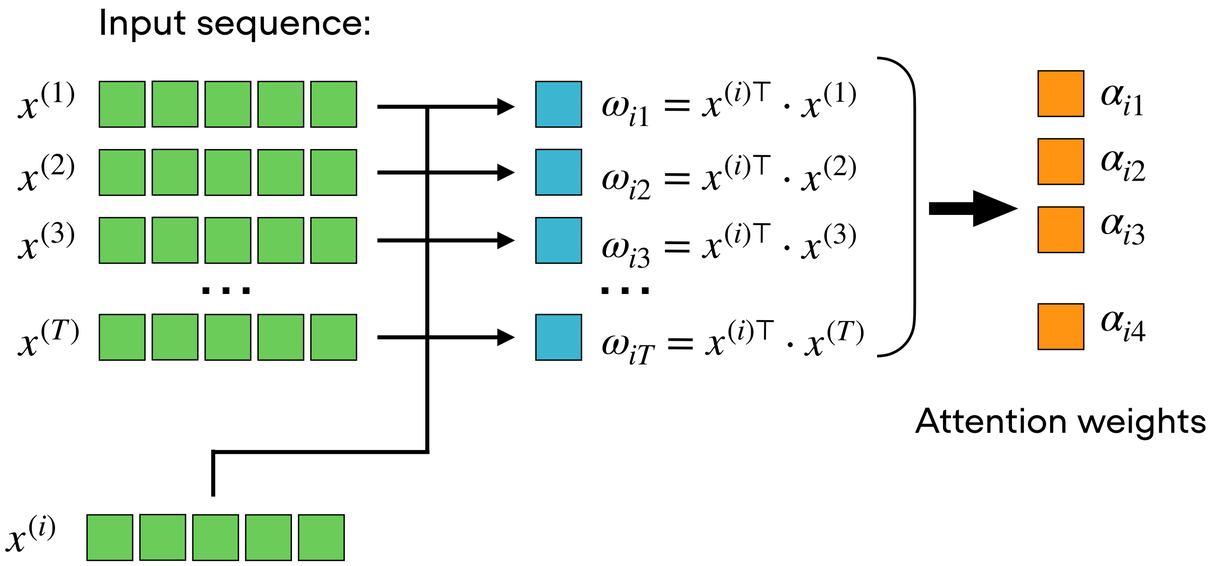
Understanding Self-Attention

Part 2: Self-Attention with Learnable Weights

Sebastian Raschka and the Lightning AI Team

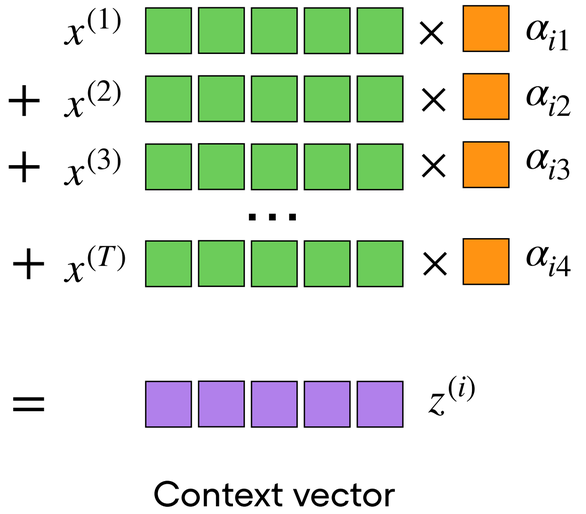
1. Compute similarity (get ω's)

2. Normalize (get α's)



3. Compute context vector $z^{(i)}$

$$z^{(i)} = \sum_{j=1}^{T} \alpha_{ij} \cdot x^{(j)}$$



A self-attention mechanism with learnable weights:
scaled dot-product attention

A self-attention mechanism with learnable weights:
scaled dot-product attention

Proposed in the original transformer paper and the most widely used attention mechanism today

We introduce 3 weight matrices

$$U_q$$

$$U_k$$

$$U_v$$

We introduce 3 weight matrices

query sequence: $q^{(i)} = U_q \, x^{(i)}$ for $i \in [1,...,T]$

key sequence: $k^{(i)} = U_k \, x^{(i)}$ for $i \in [1,...,T]$

value sequence: $v^{(i)} = U_v \, x^{(i)}$ for $i \in [1,...,T]$

Query, key, and value are inspired by databases
(and information retrieval systems).

If we enter a query, it is matched against a key to retrieve certain values.

Similar to the previous lecture (basic self-attention)
it's about computing a context vector.

Input sequence:

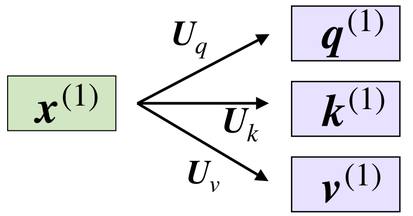
$\boldsymbol{x}^{(1)}$

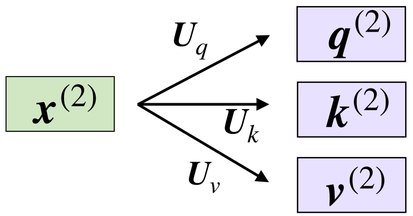
$\boldsymbol{x}^{(2)}$

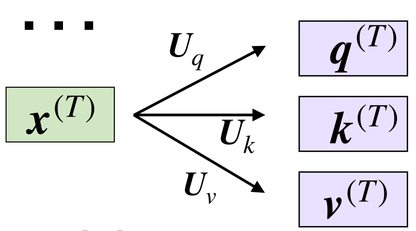
...

$\boldsymbol{x}^{(T)}$

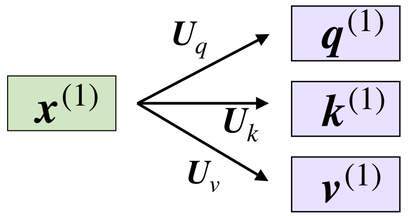
1. Compute **key**, **query**, and **value** vectors (vector-matrix multiplication)

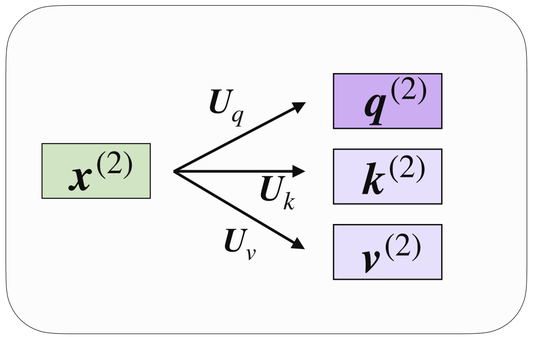


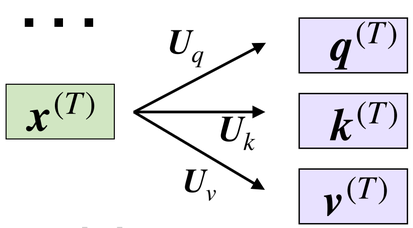




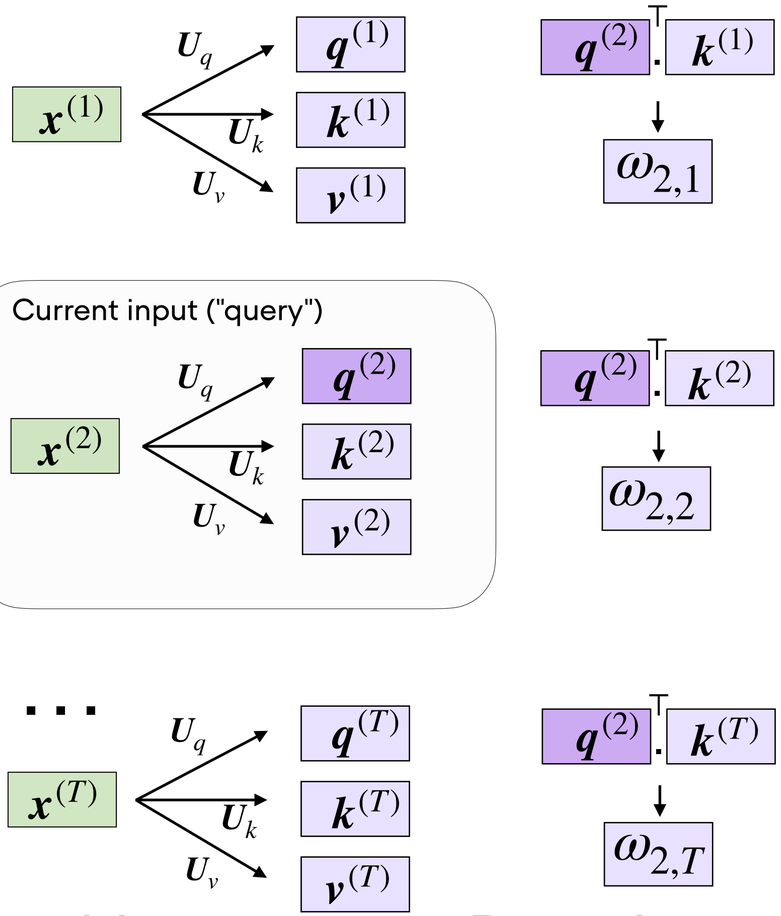
Suppose we want to compute the context vector for the **2nd input element**

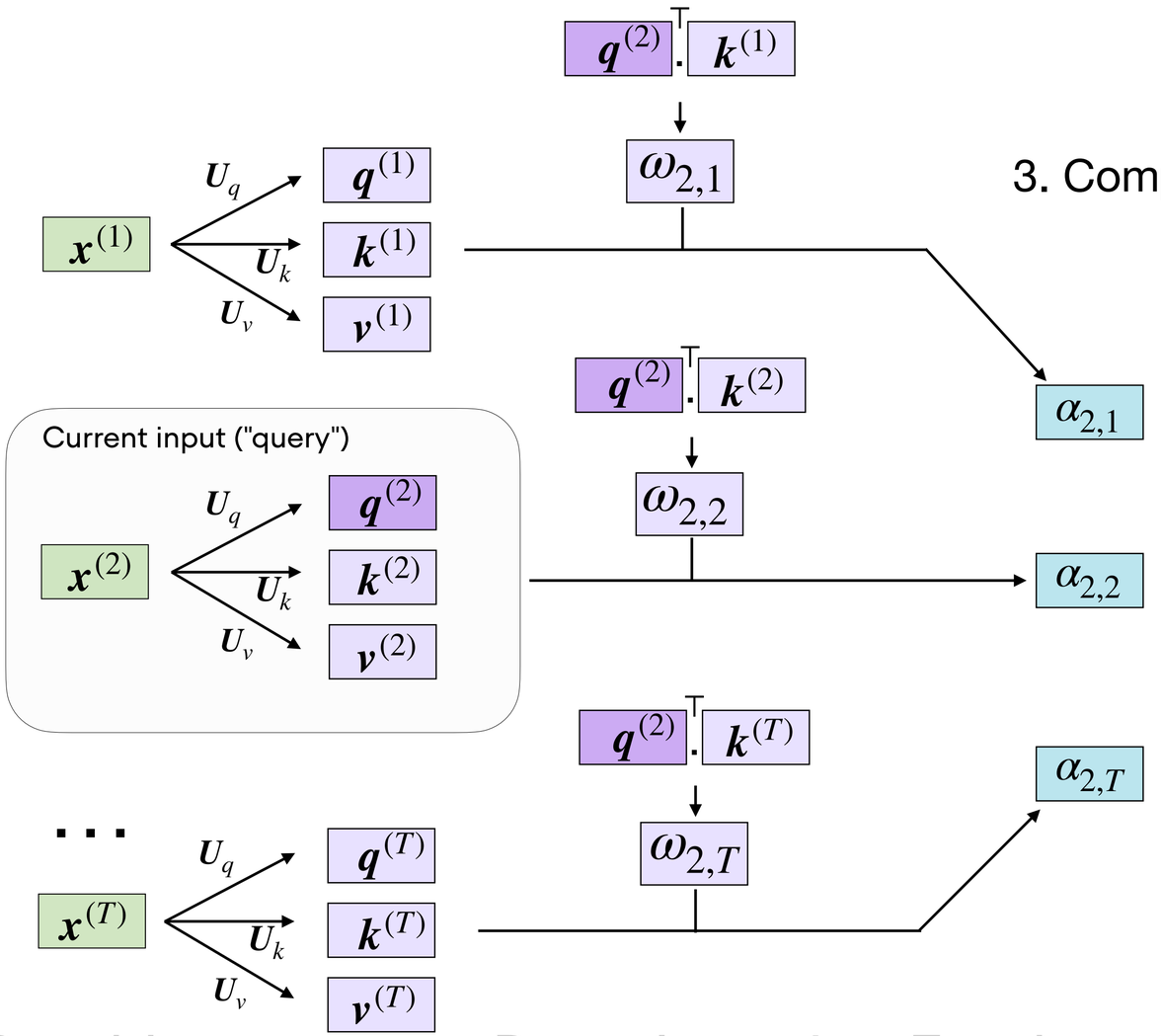






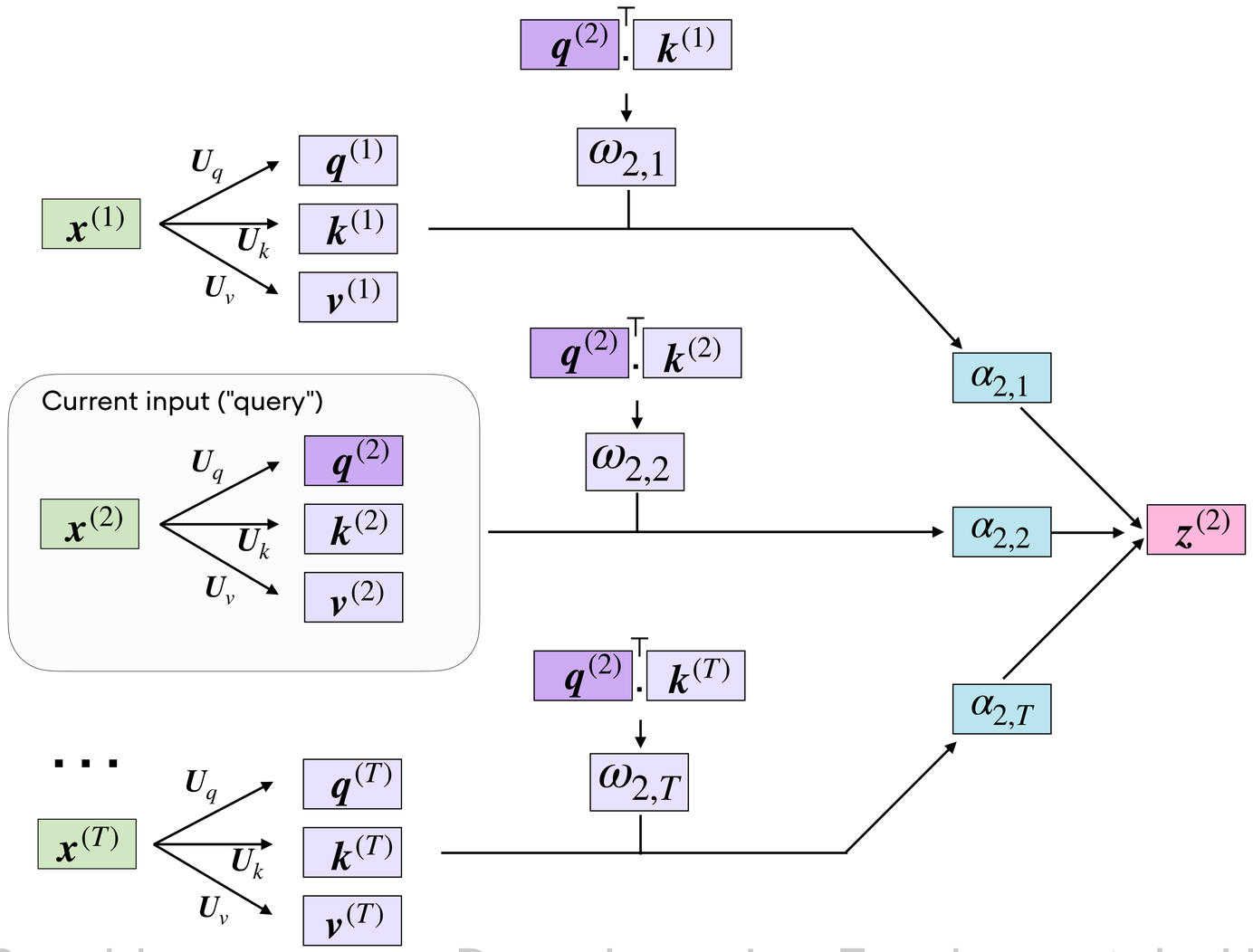
2. Compute ω (similarity) as in previous video (but now between q's and k's instead of x's)





3. Compute attention scores α (by normalizing ω's)

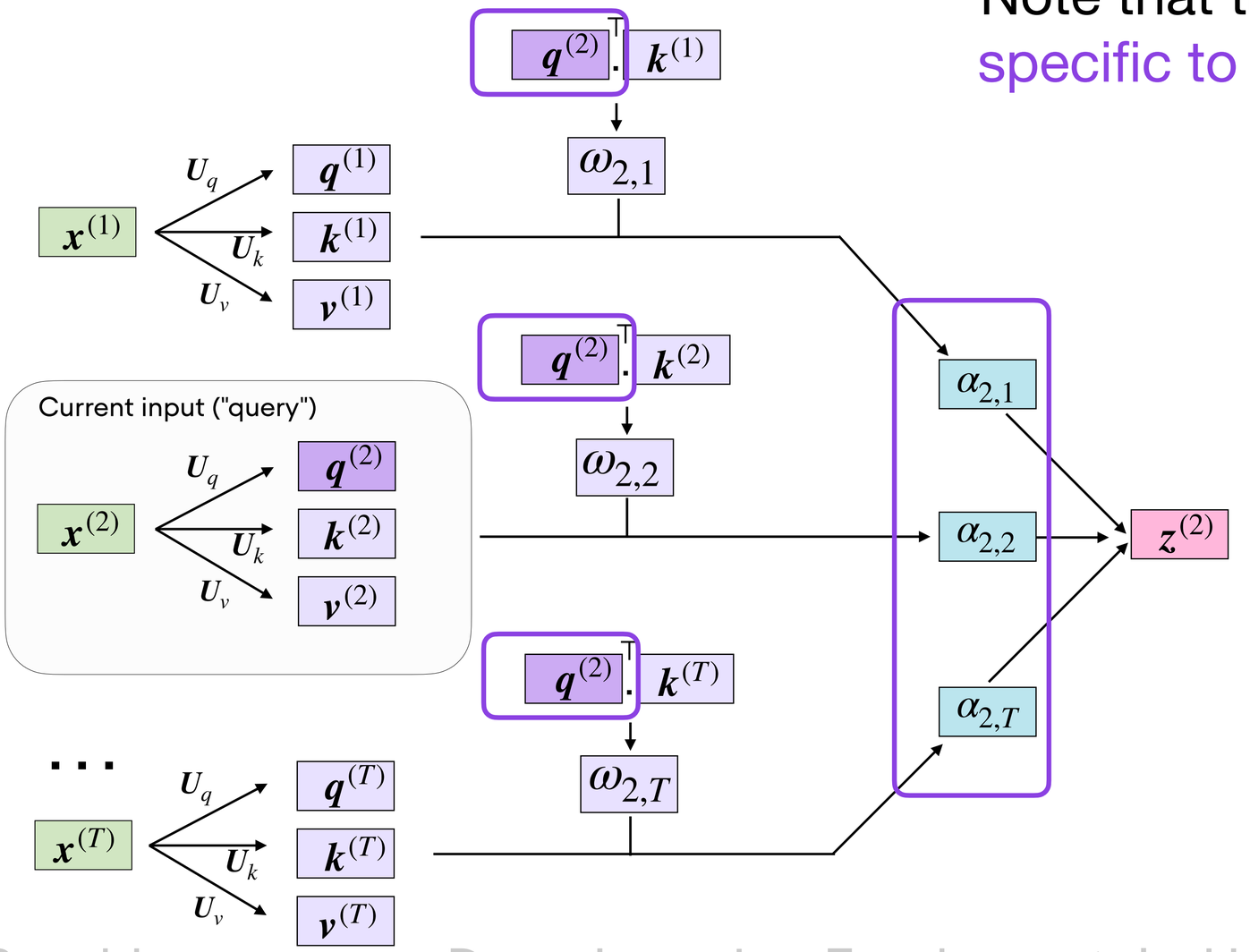
where $\alpha_{2,i} = \text{softmax}\left(\frac{\omega_{2,i}}{\sqrt{d_k}}\right)$



4. Compute context vector $z^{(2)}$

where $z^{(2)} = \sum_{j=1}^{T} \alpha_{2,j} \, v^{(j)}$

Note that the attention scores are specific to the current input token



where $z^{(2)} = \sum_{j=1}^{T} \alpha_{2,j} v^{(j)}$

Summary:

for each token, self attention

- compares that token to each other token in the input sequence
- computes attention scores unique to the input token (query)
- calculates the weighted average of all inputs via the attention scores

Self-attention is a sequence-to-sequence (many-to-many) approach:

- taking n tokens as input and
- returning n tokens as output.

This scaled dot-product attention mechanism is used in the multi-head attention blocks

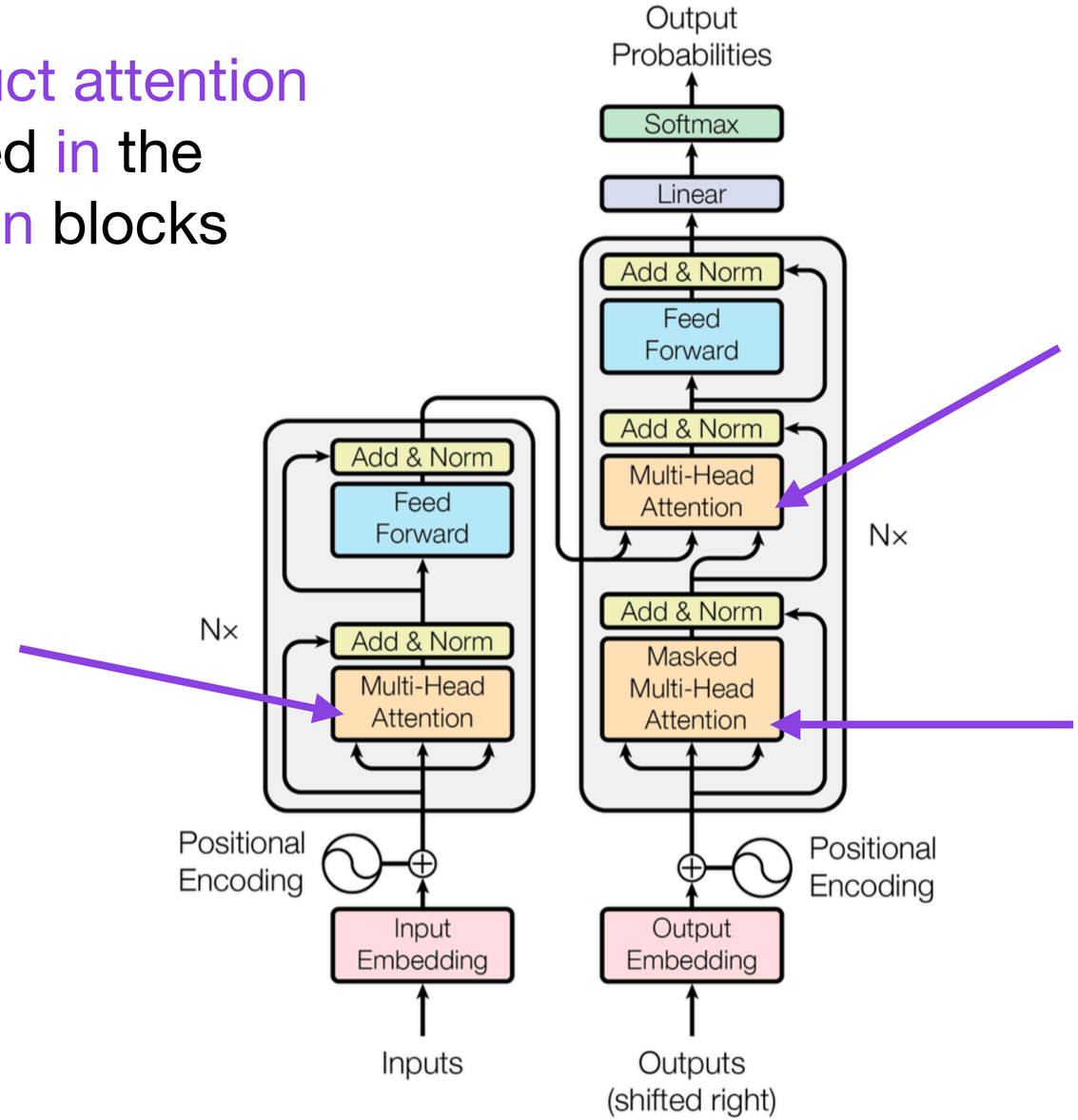


Figure 1: The Transformer - model architecture.

Next: What is “multi-head” attention?