香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

COMP 4901B

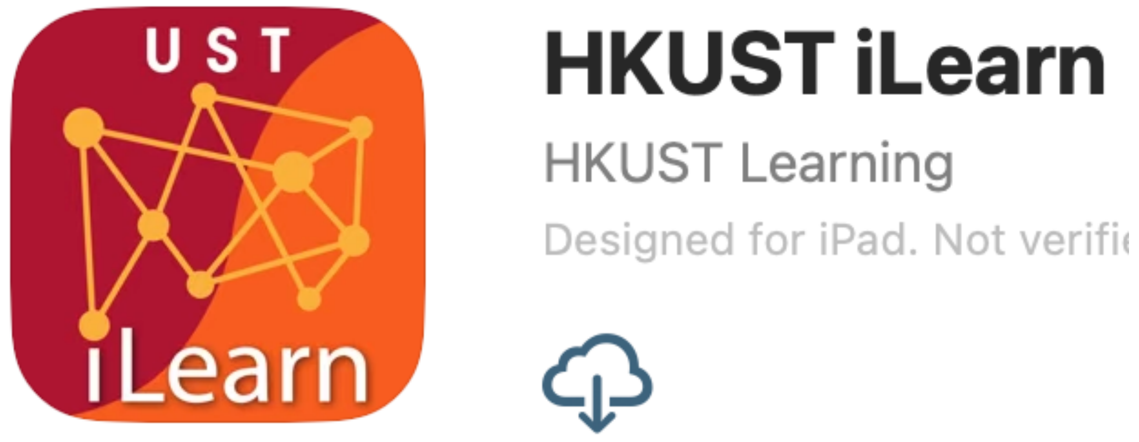
Large Language Models

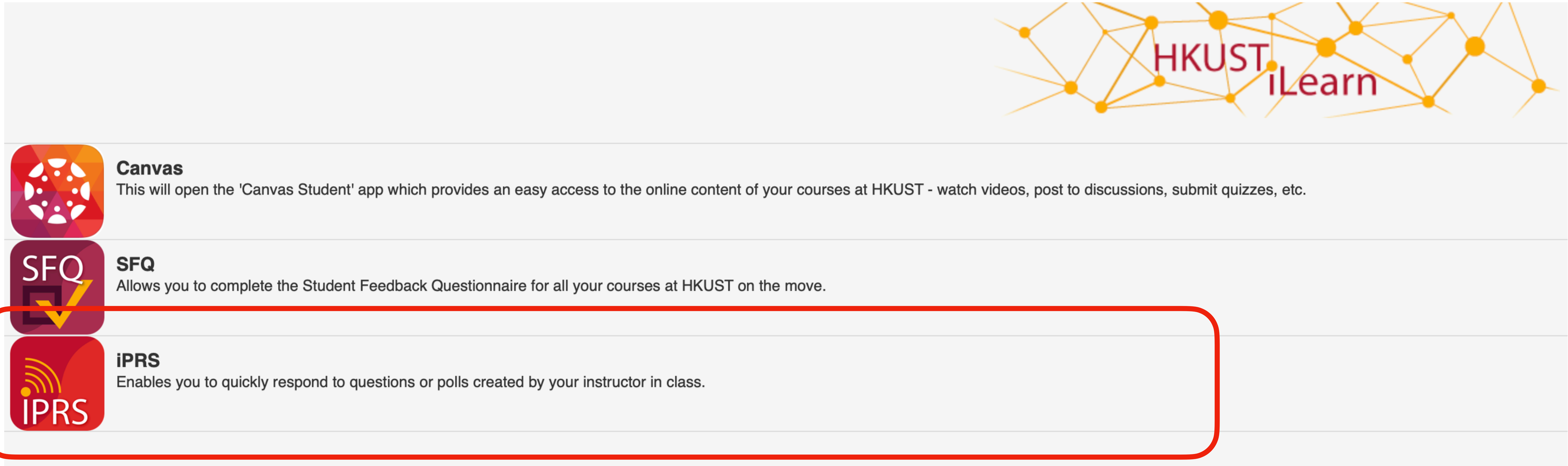
Transformers

Junxian He

Sep 17, 2025

Please Download HKUST iLearn in Your Mobile Phone or iPad





We are going to use iPRS to do quizzes in the future

Recap: Autoregressive Language Models

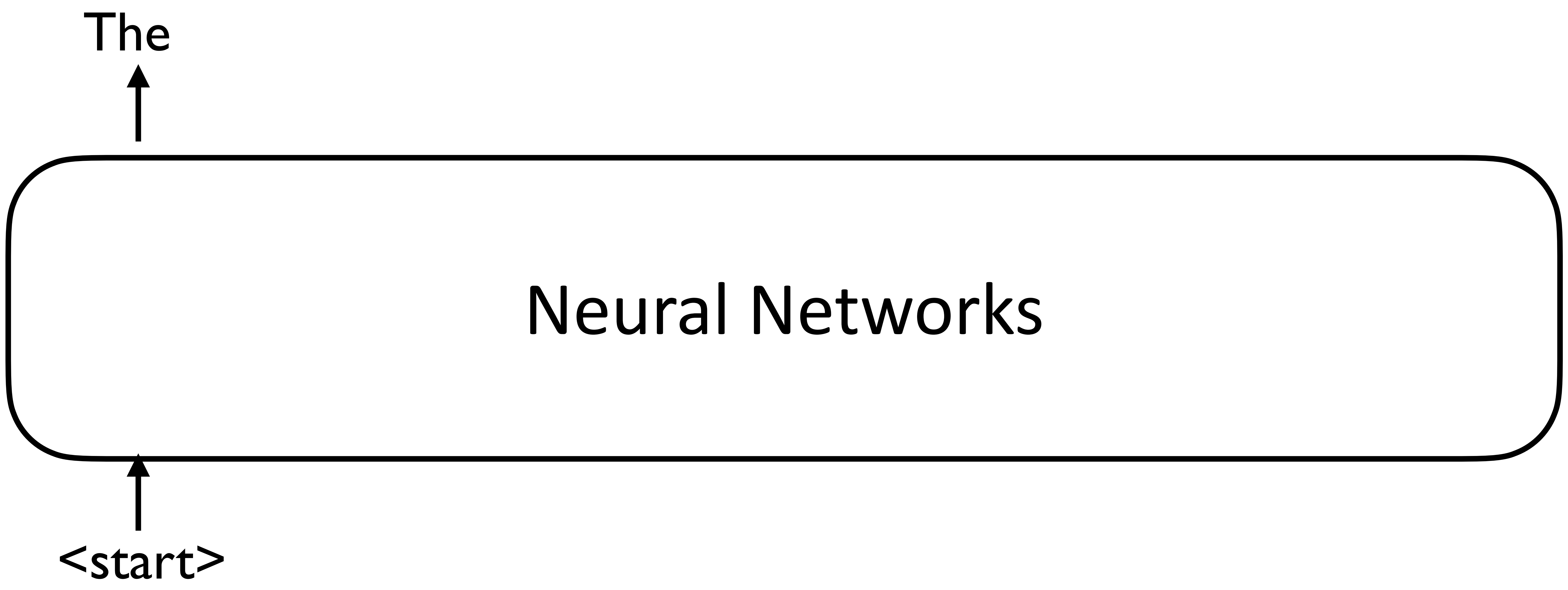
$$
\begin{aligned}
p(\text{the, mouse, ate, the, cheese}) = p(&\text{the}) \\
p(&\text{mouse} \mid \text{the}) \\
p(&\text{ate} \mid \text{the, mouse}) \\
p(&\text{the} \mid \text{the, mouse, ate}) \\
p(&\text{cheese} \mid \text{the, mouse, ate, the}).
\end{aligned}
$$

$$p(x_1, x_2, \ldots, x_I) = \prod_{i=1}^{I} p(\underbrace{x_i}_{\text{Next Word}} \mid \underbrace{x_{1:i-1}}_{\text{Context}})$$

Recap: Neural Language Models

Neural language models are typically autoregressive

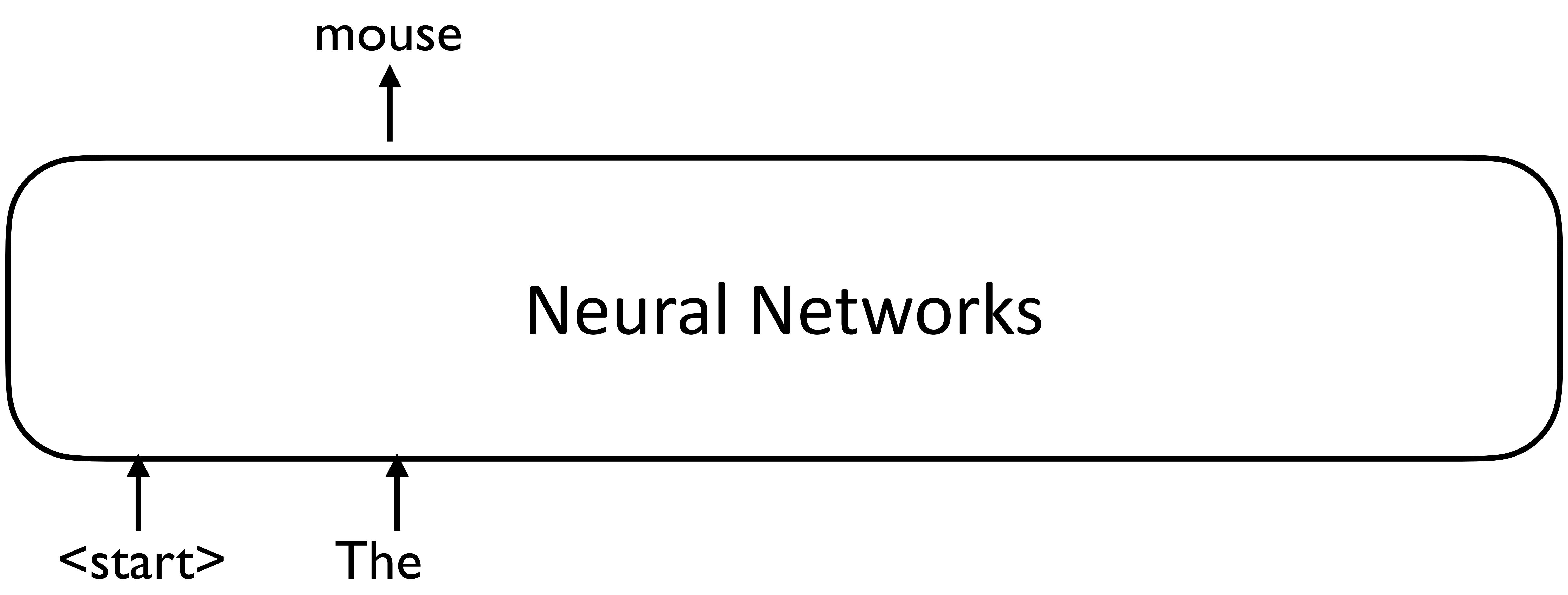
Data: “The mouse ate the cheese .”



Recap: Neural Language Models

Neural language models are typically autoregressive

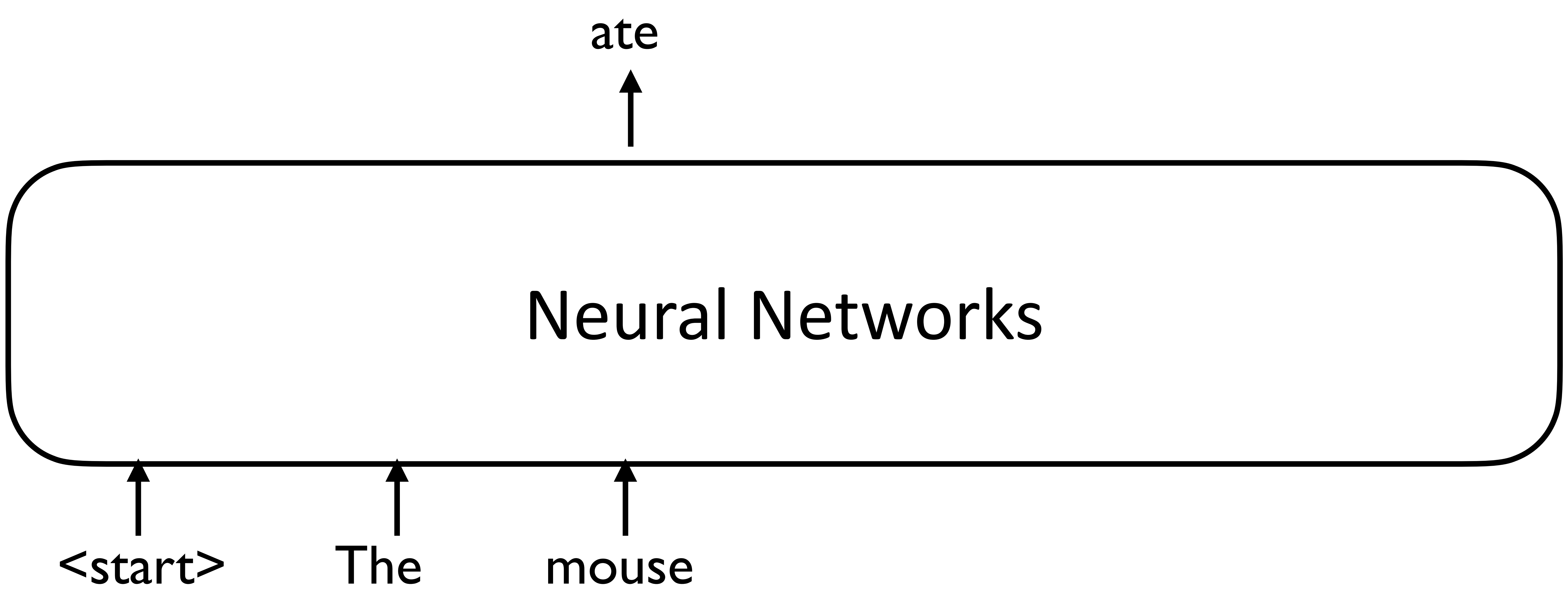
Data: "The mouse ate the cheese ."



Recap: Neural Language Models

Neural language models are typically autoregressive

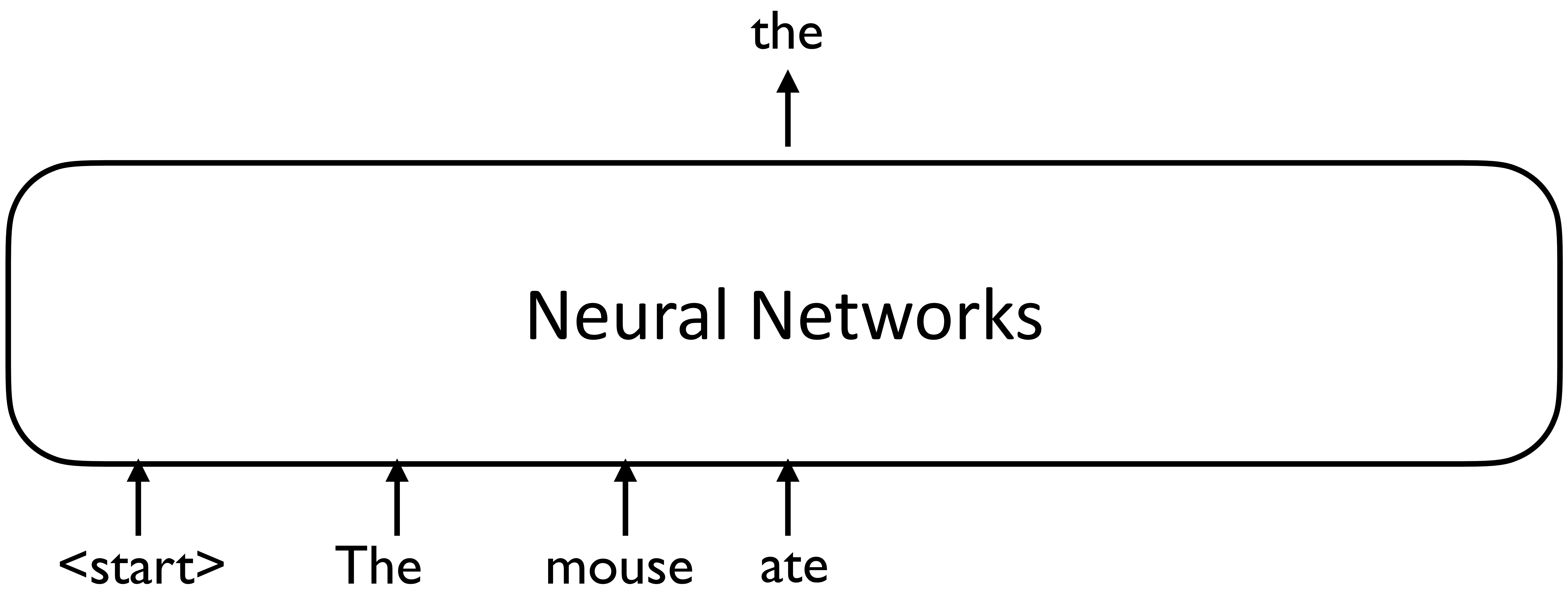
Data: "The mouse ate the cheese ."



Recap: Neural Language Models

Neural language models are typically autoregressive

Data: "The mouse ate the cheese ."

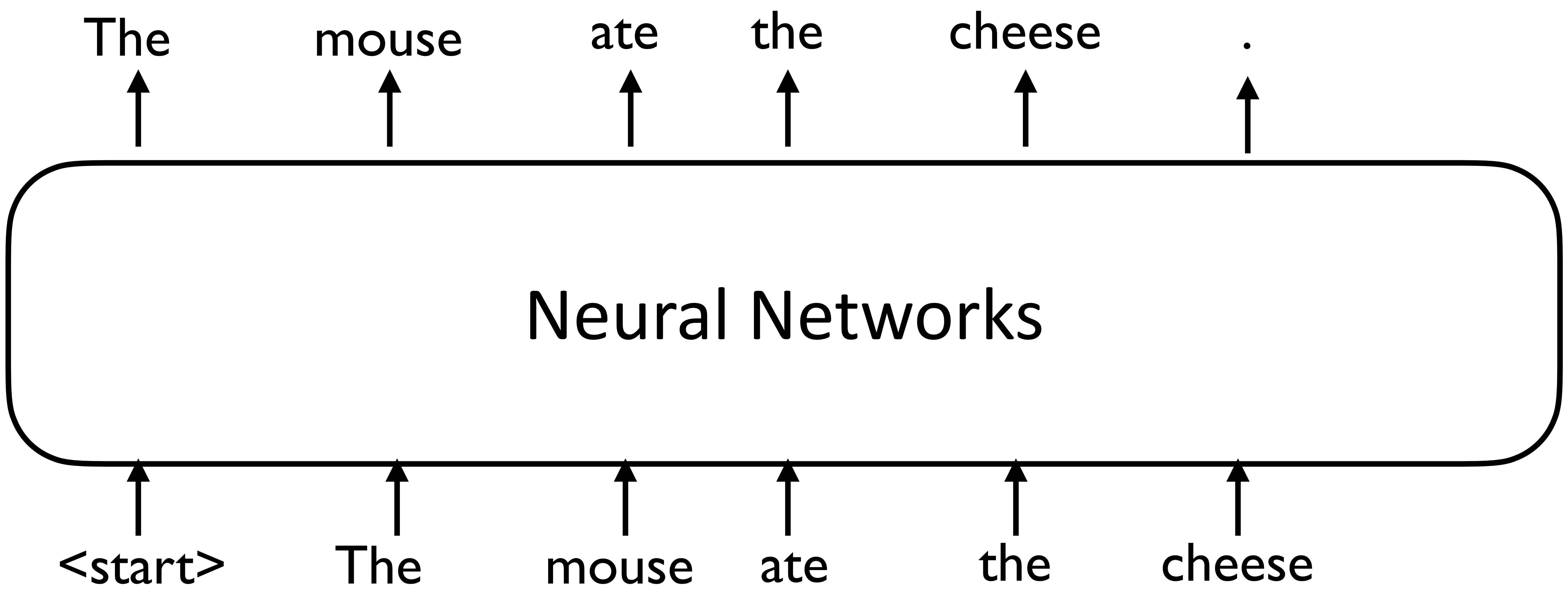


We can compute the loss on every token in parallel

Recap: Neural Language Models

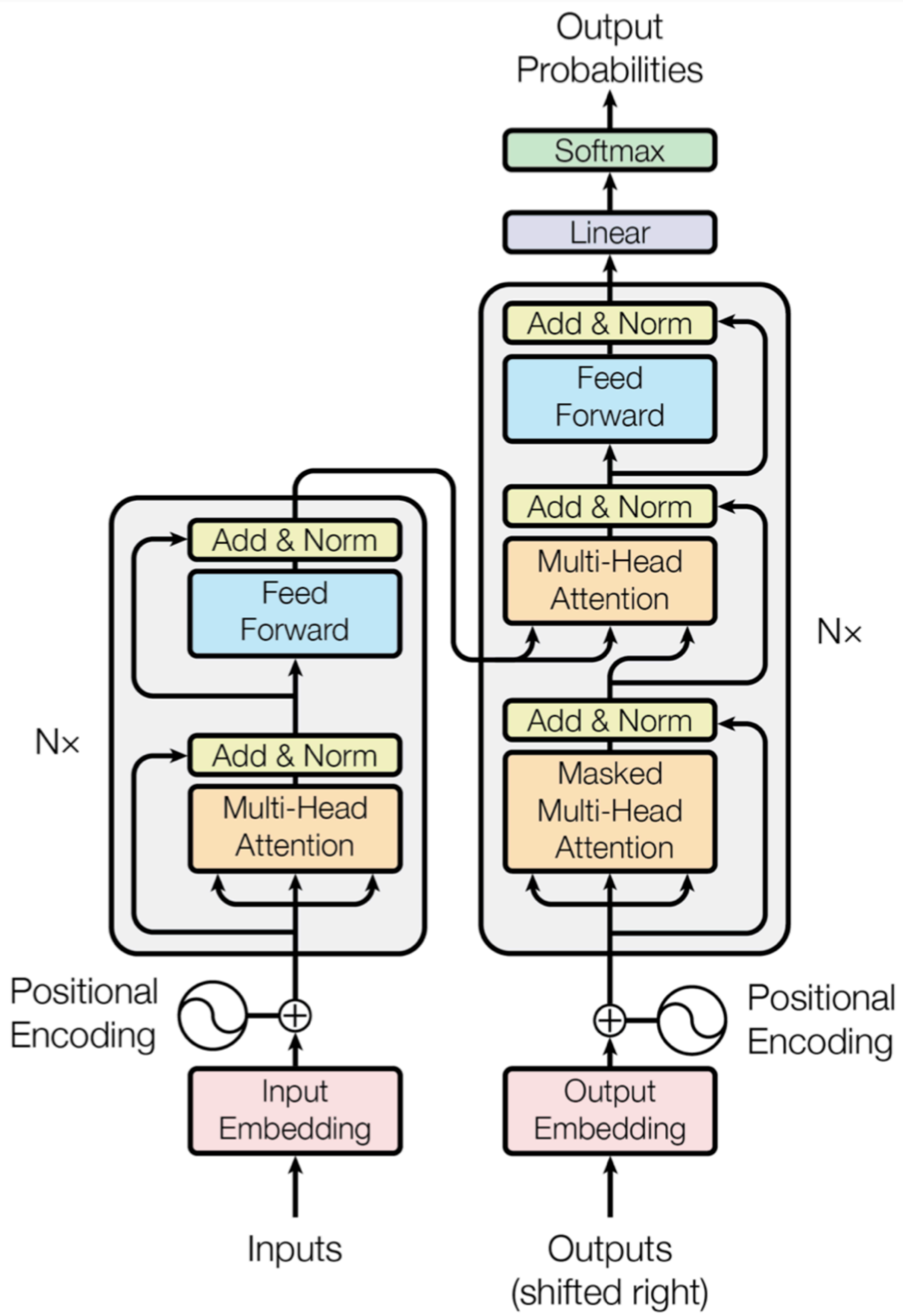
Neural language models are typically autoregressive

Data: "The mouse ate the cheese ."



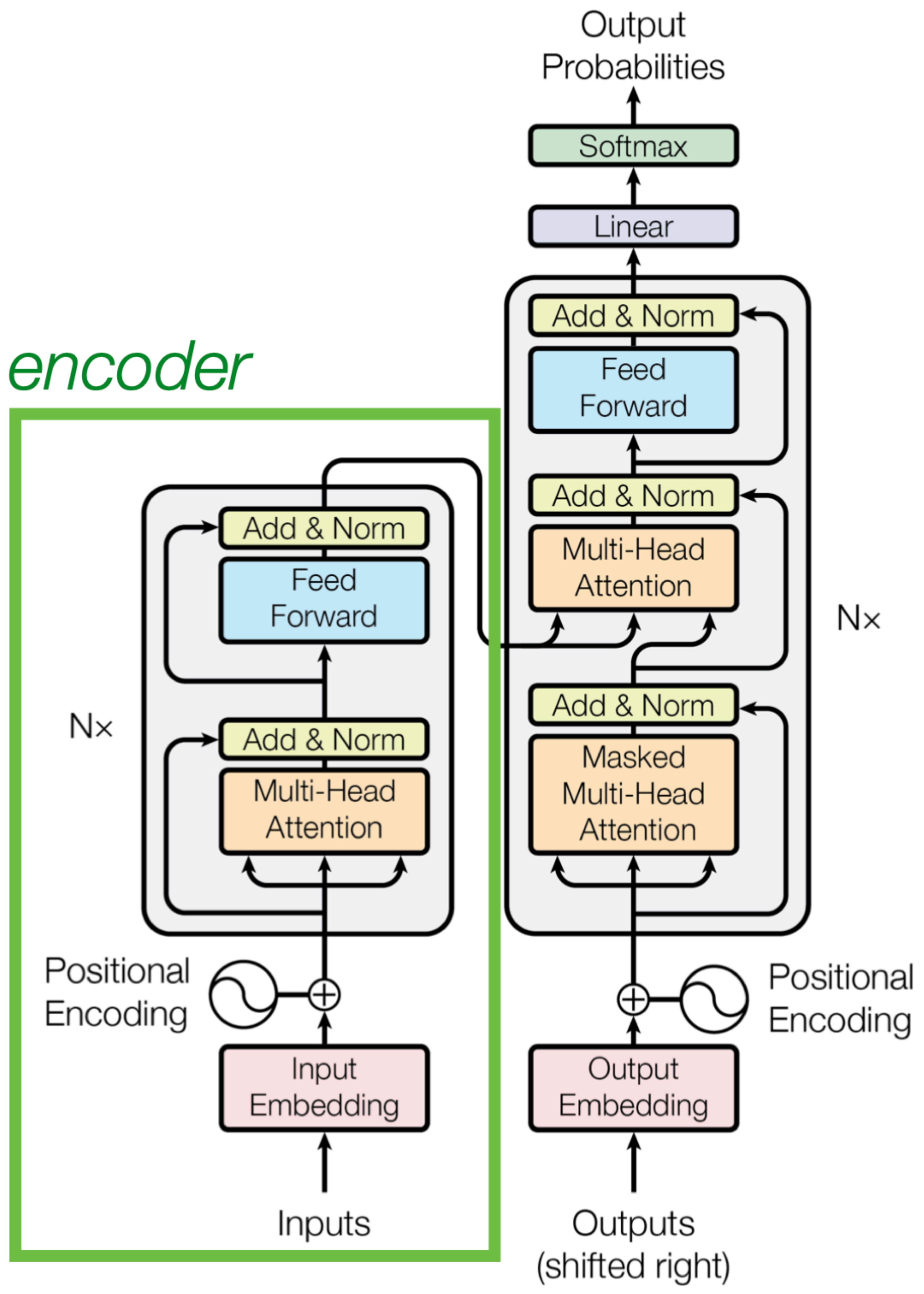
Each prediction only sees the inputs on its left

Recap: Transformer

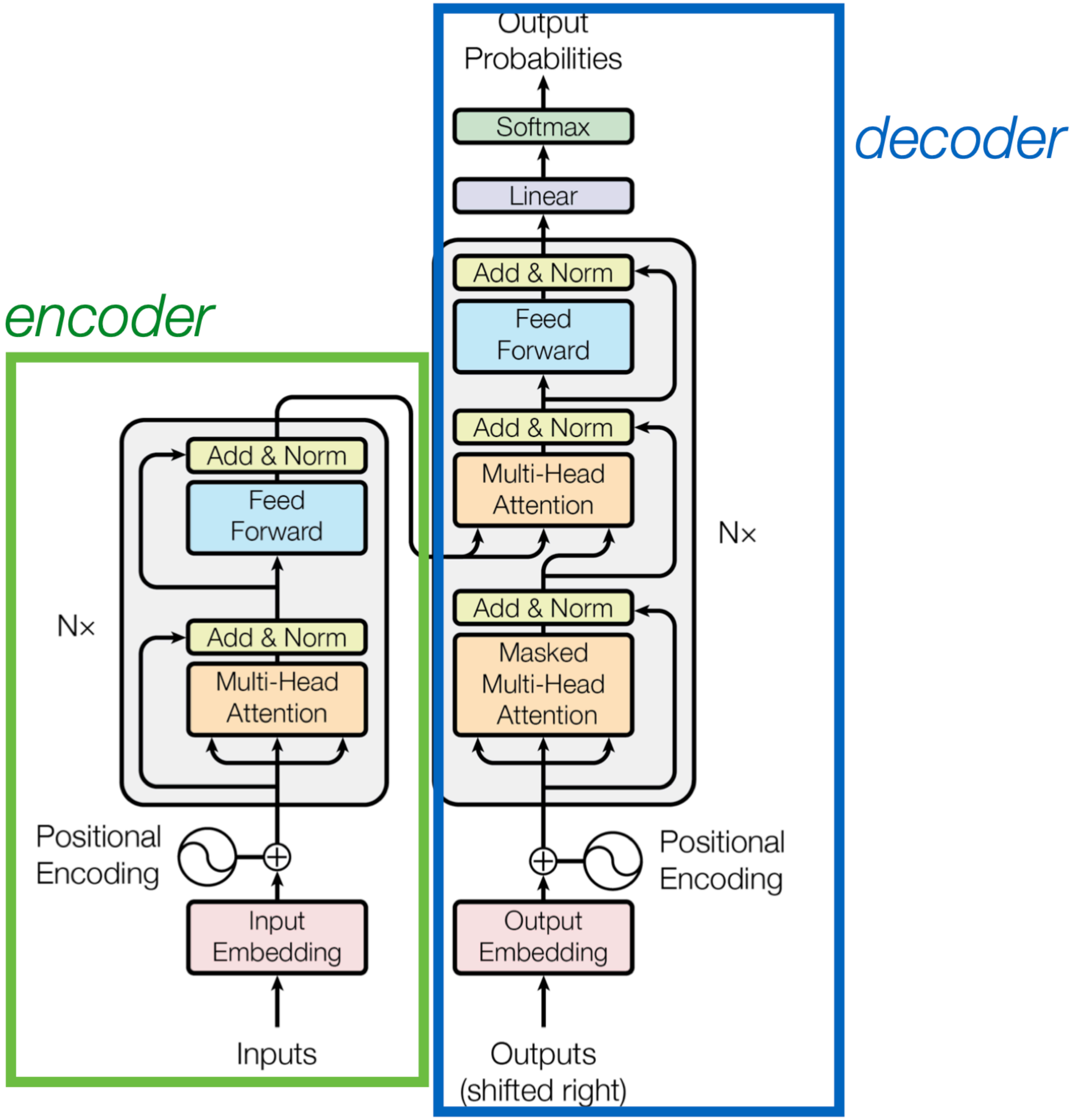


Vaswani et al. Attention is All You Need. NeurIPS 2017.

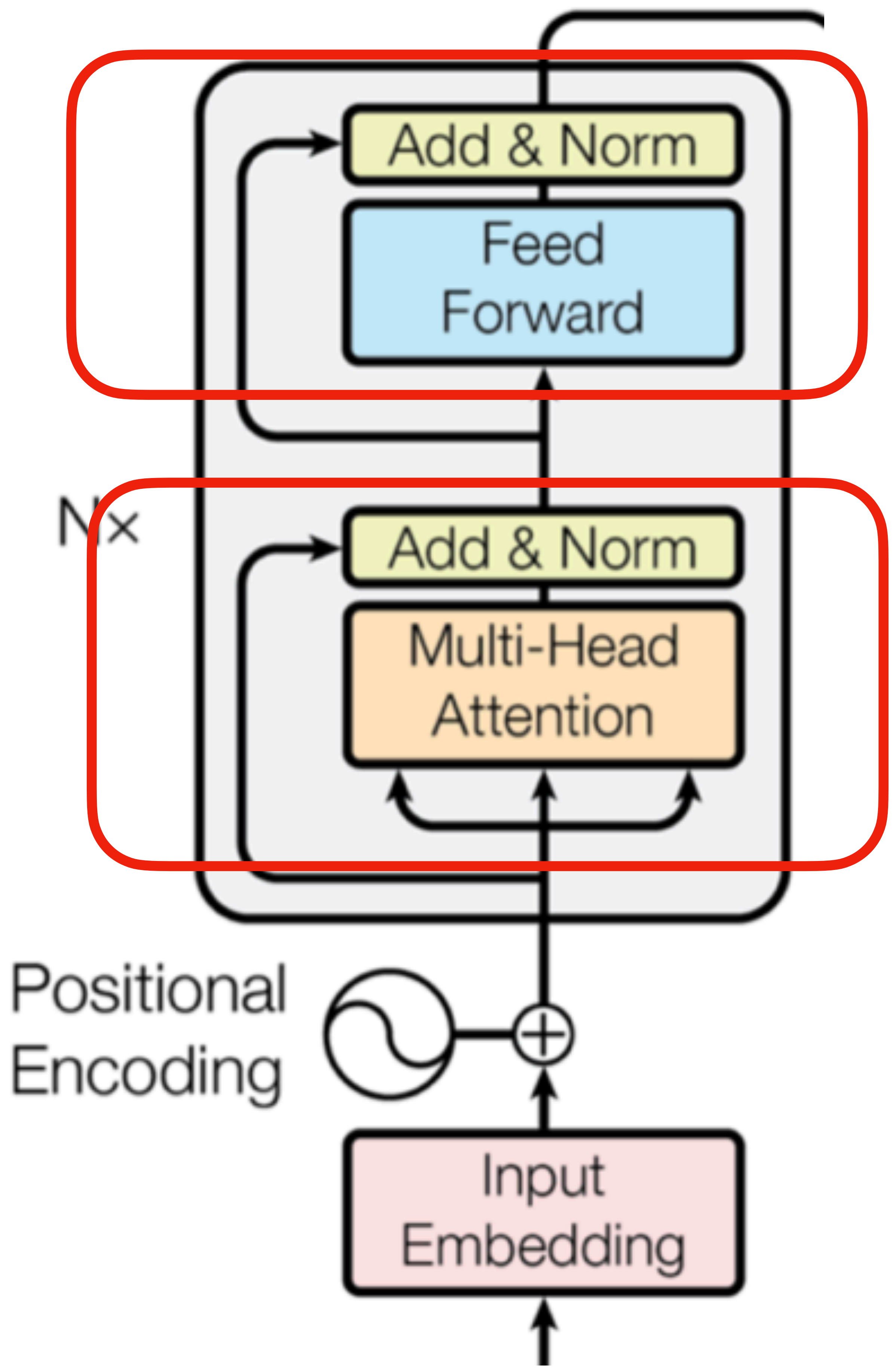
Recap: Encoder



Recap: Decoder



Recap: Transformer Encoder



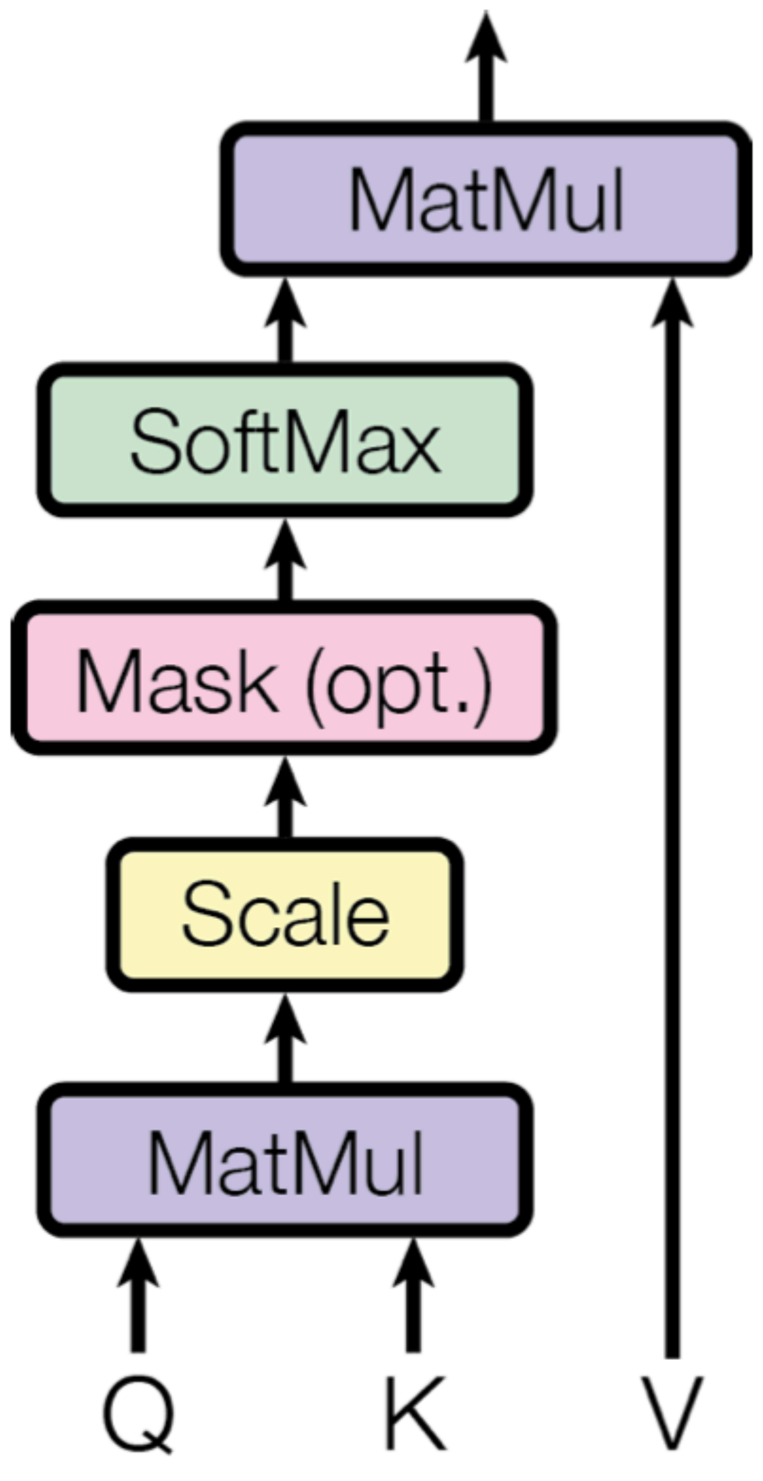
Residual connection

MLP

Self-attention

Recap: What is Attention

Scaled Dot-Product Attention



Q: Query
K: key
V: value

$Q \in R^{n \times d}$ $\quad K \in R^{m \times d}$ $\quad V \in R^{m \times d}$

We have n queries, m (key, value) pairs

Attention weight = softmax(QK^T)

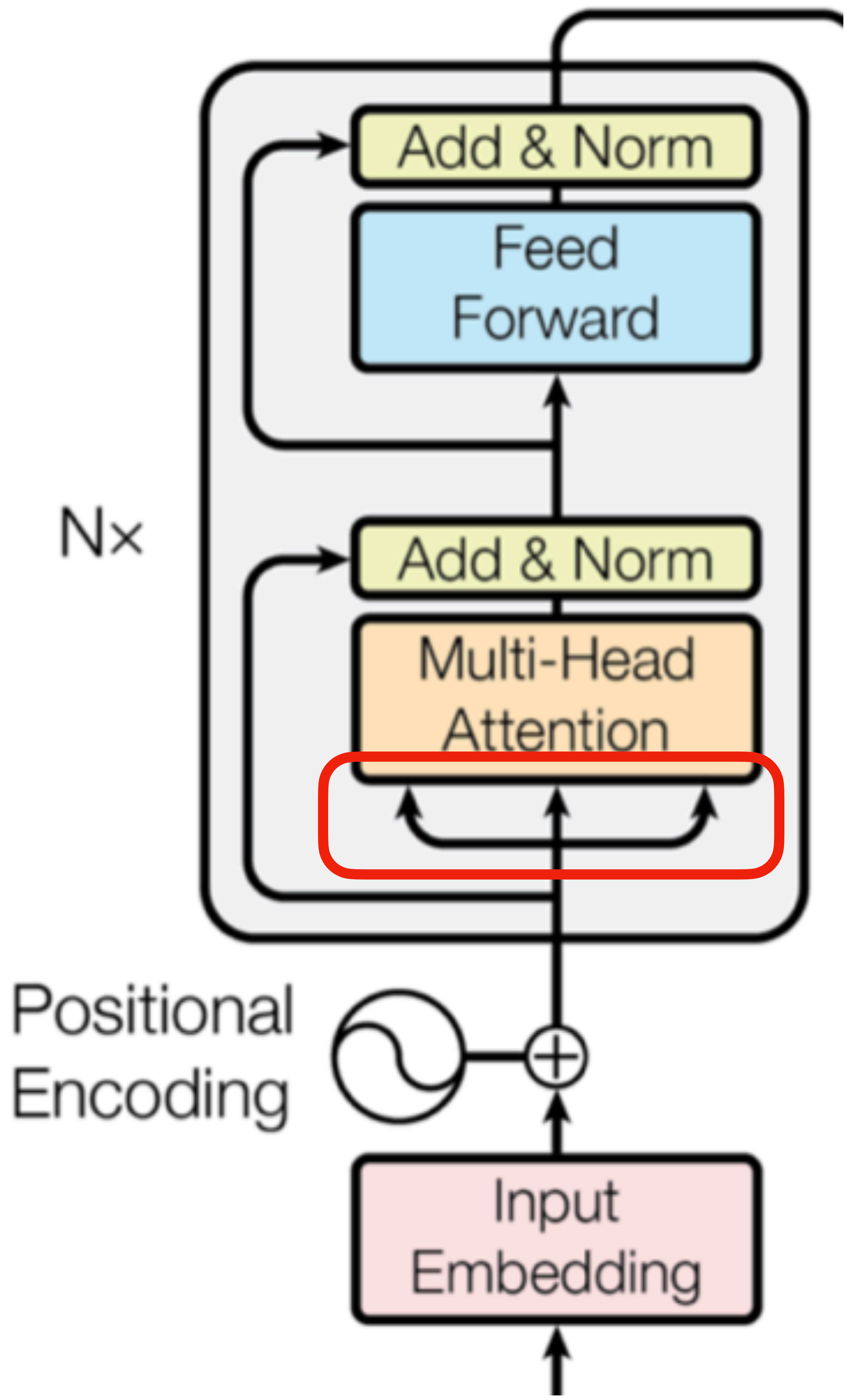
Dot-products grow large in magnitude

Scaled Attention weight = softmax($\frac{QK^T}{\sqrt{d_k}}$) Shape is mxn

Attention weight represents the strength to "attend" values V

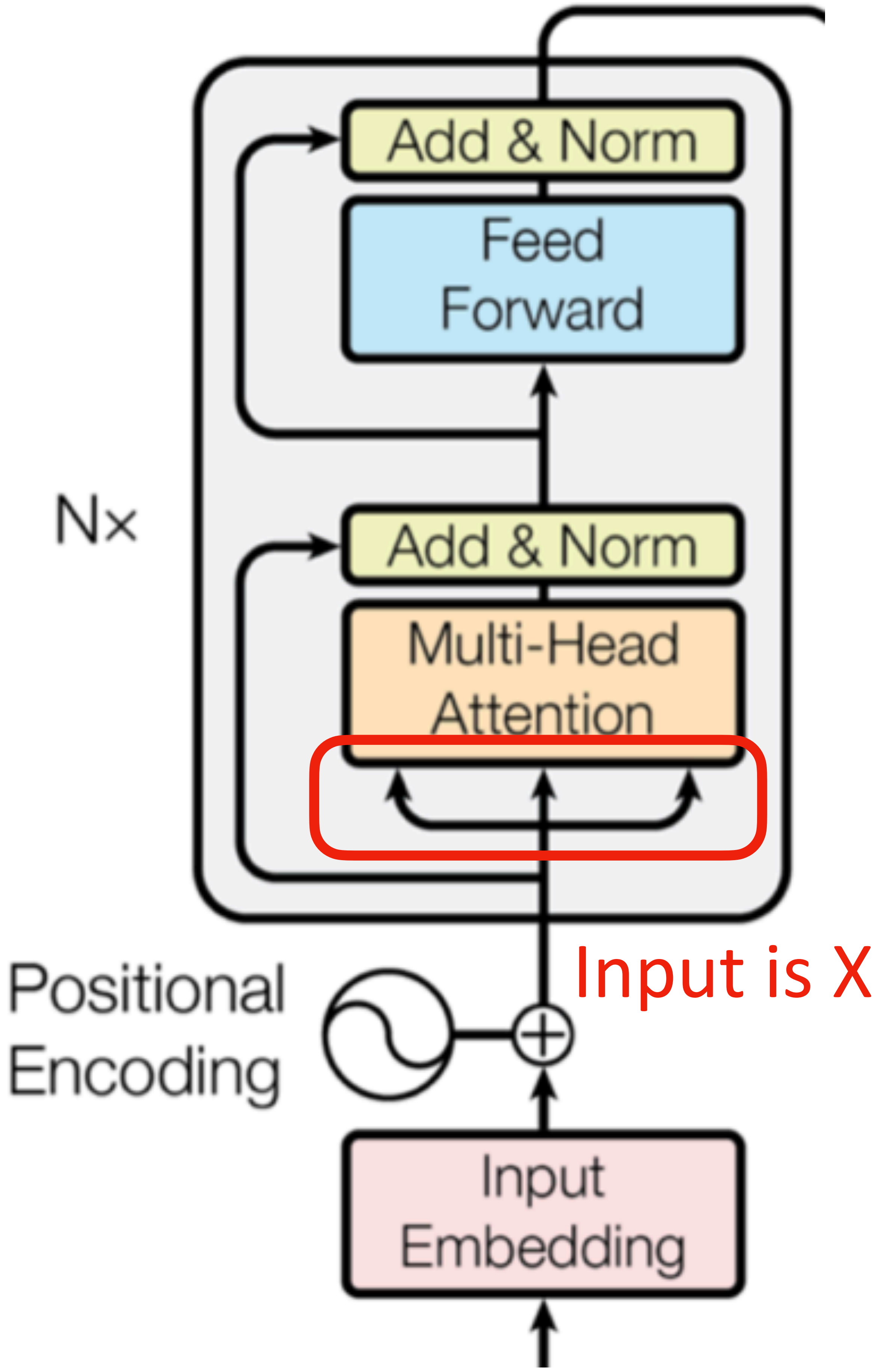
$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Q, K, V

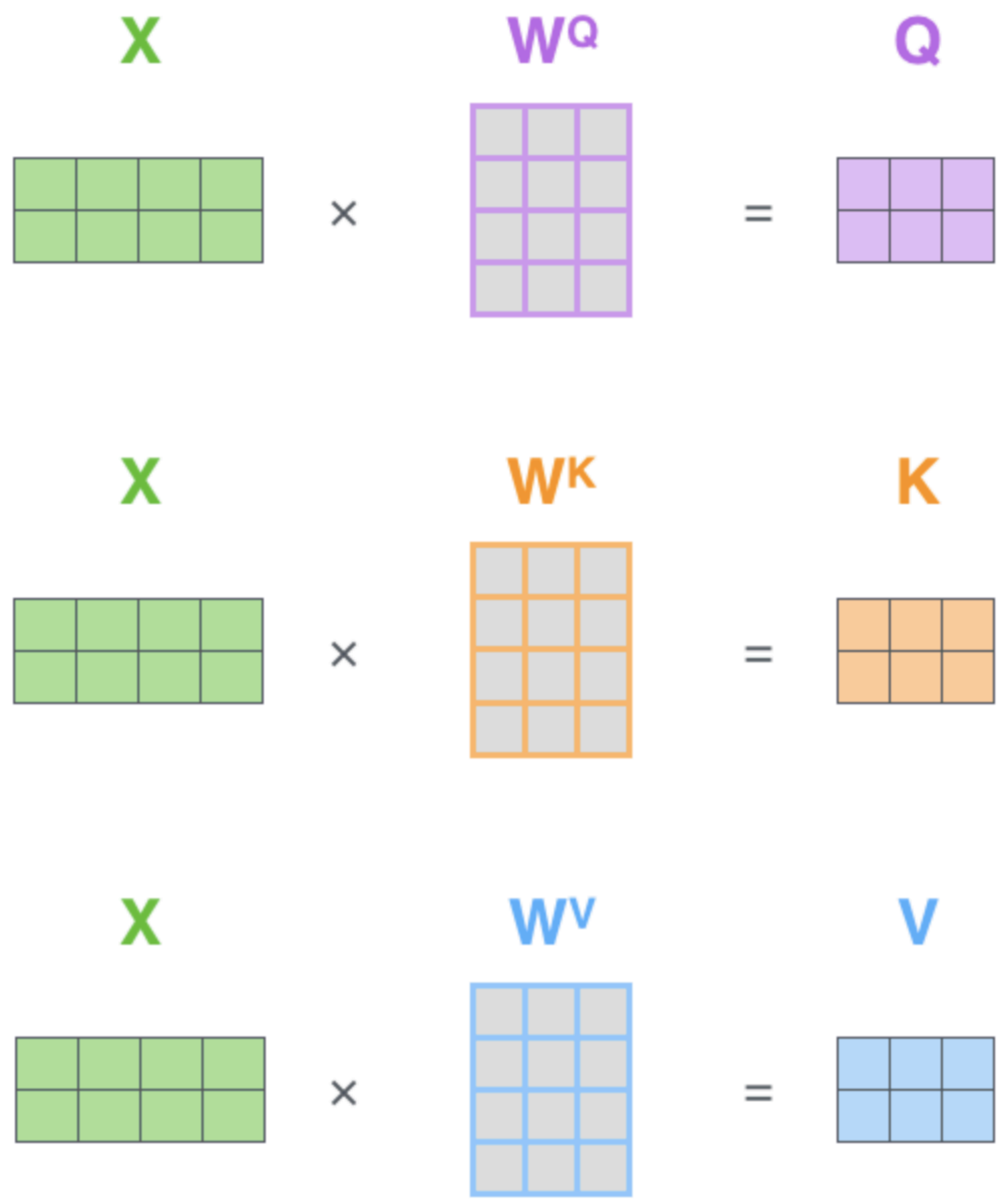


What are Q, K, V in the transformer

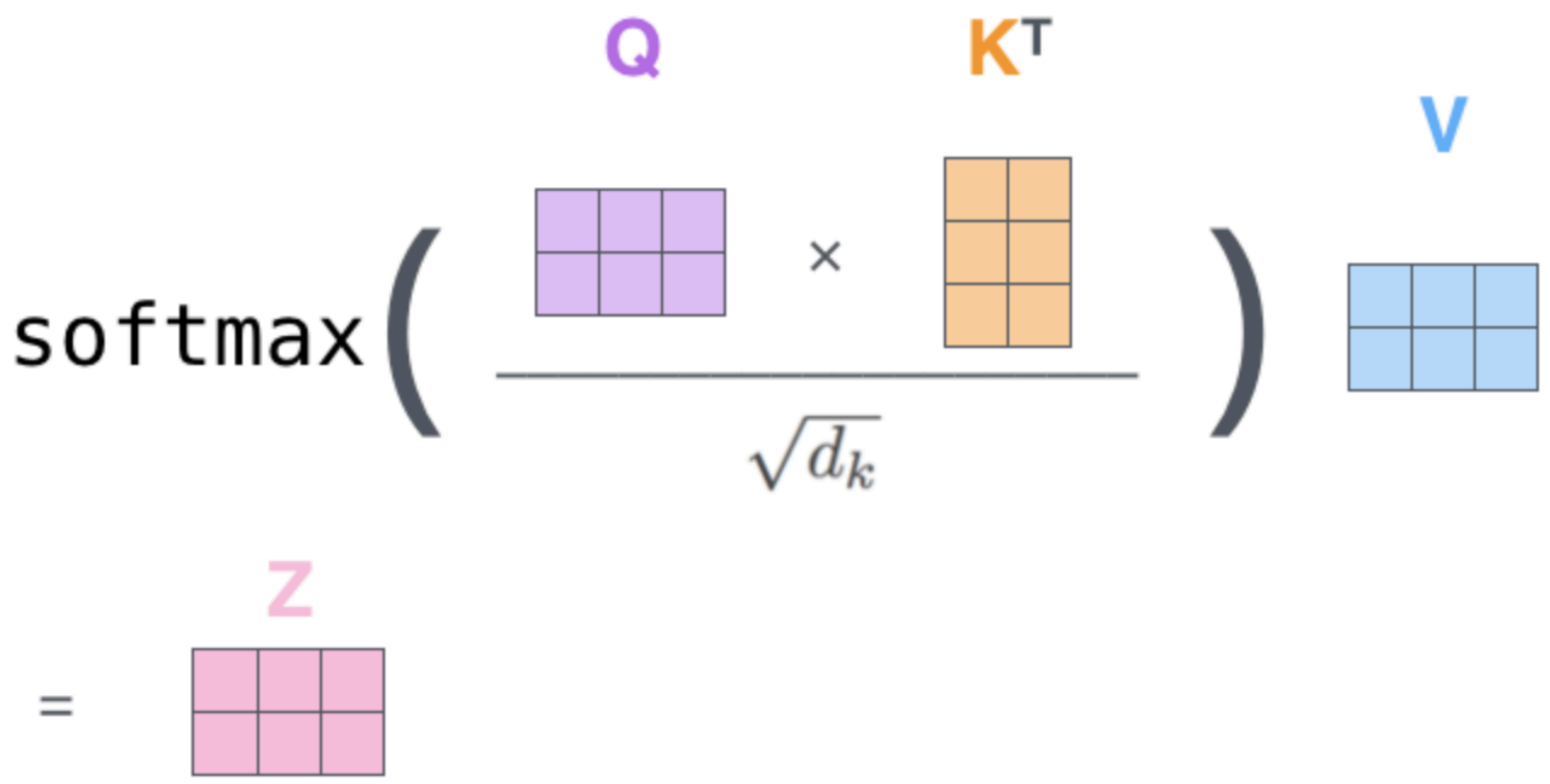
Self-Attention



15



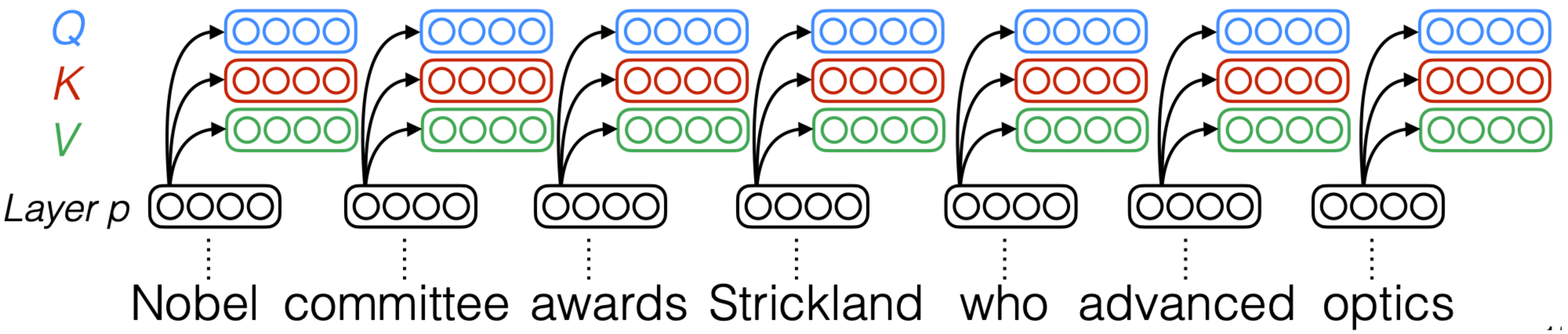
Query, key, and value are from the same input, thus it is called “self”-attention



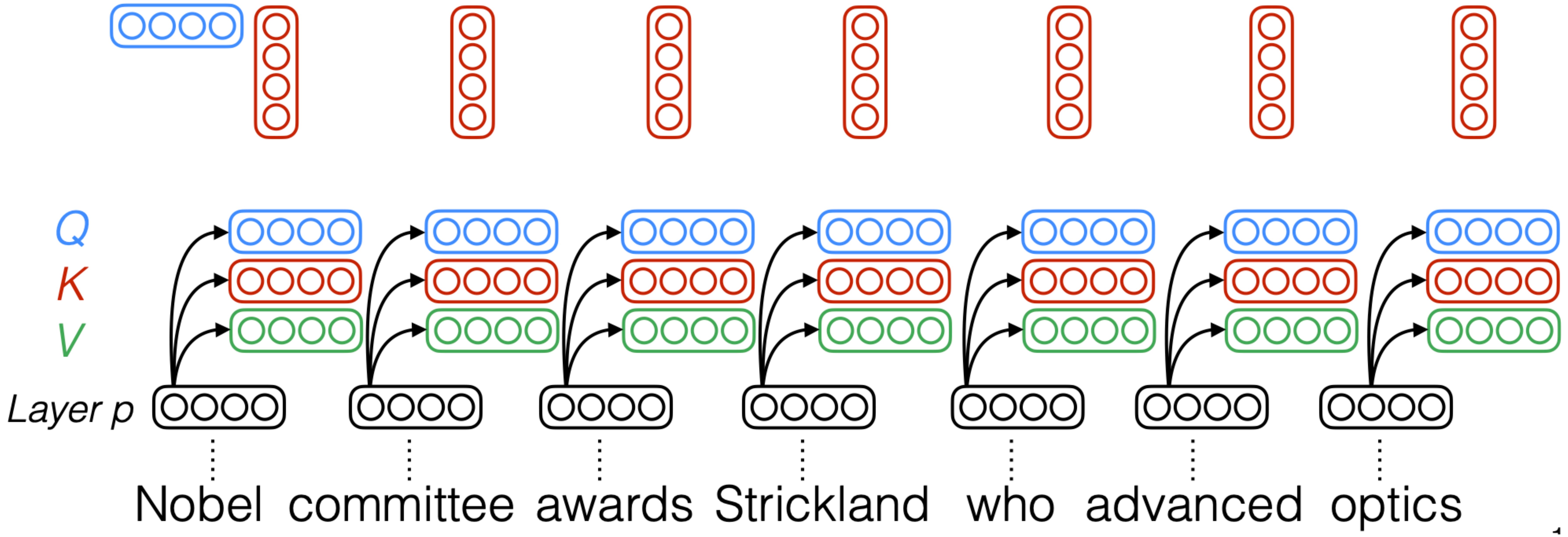
Jay Alammar. The Illustrated Transformer.

Self-Attention

At each step, the attention computation attends to all steps in the input example

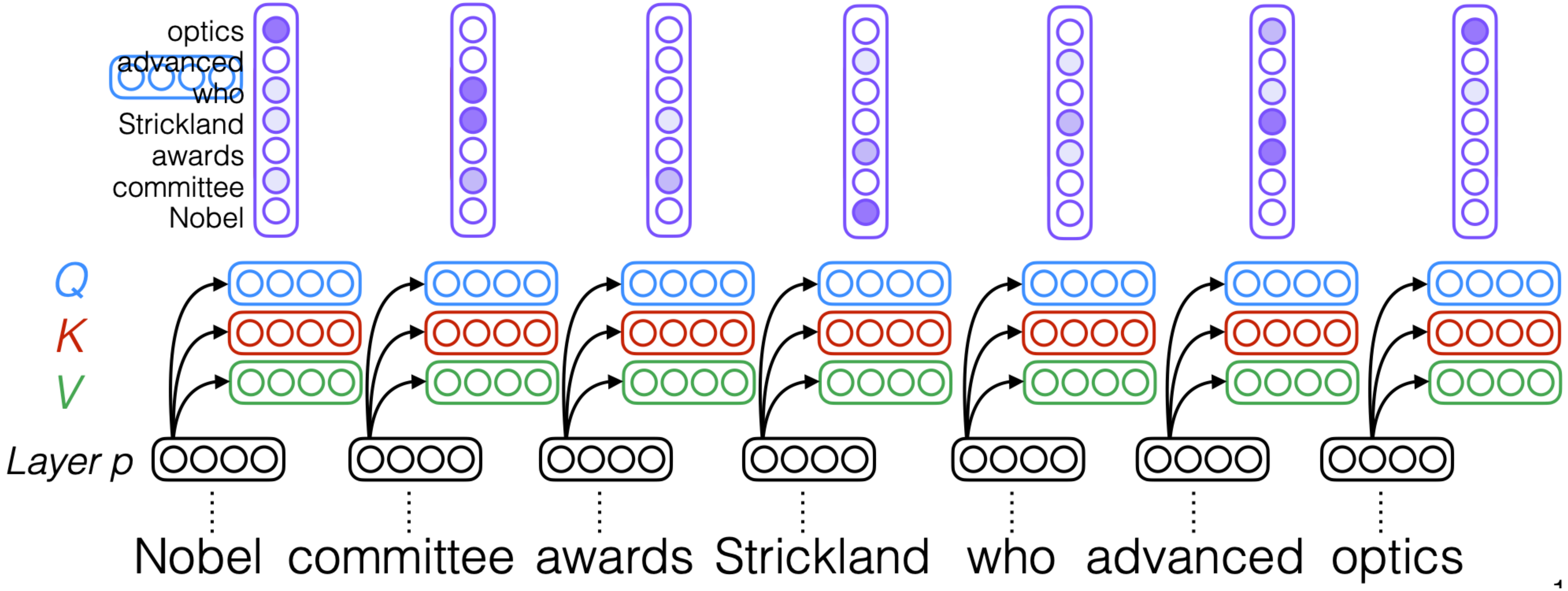


Self-Attention

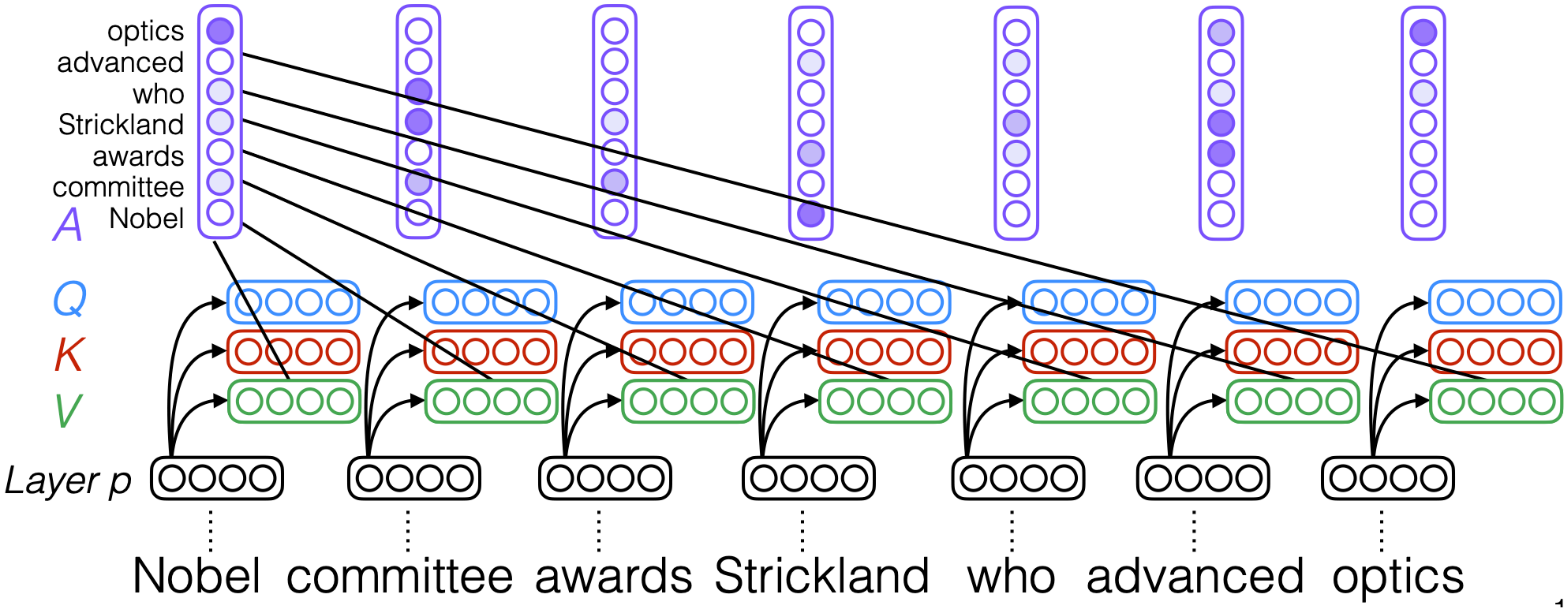


Self-Attention

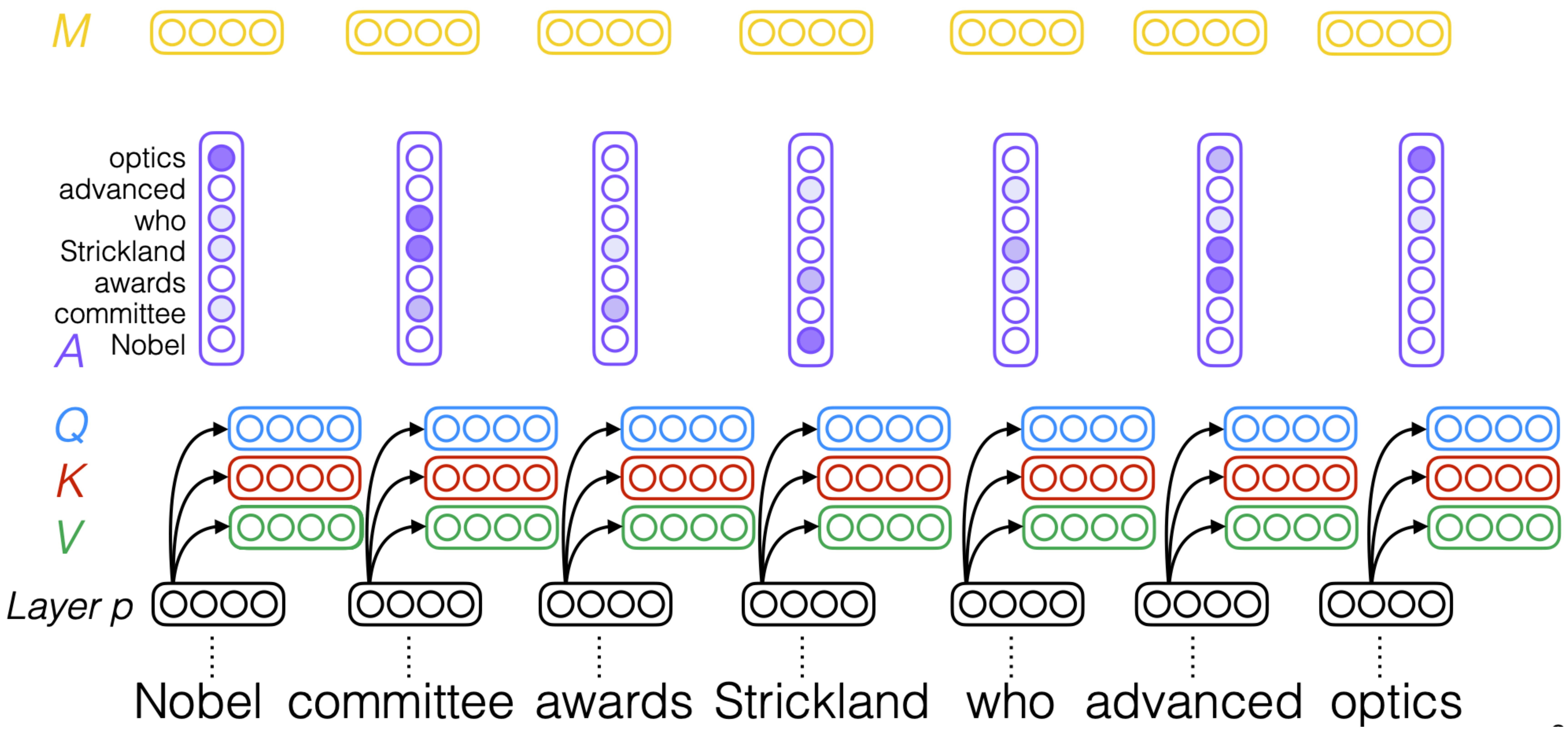
Attention weight on every word in the sequence



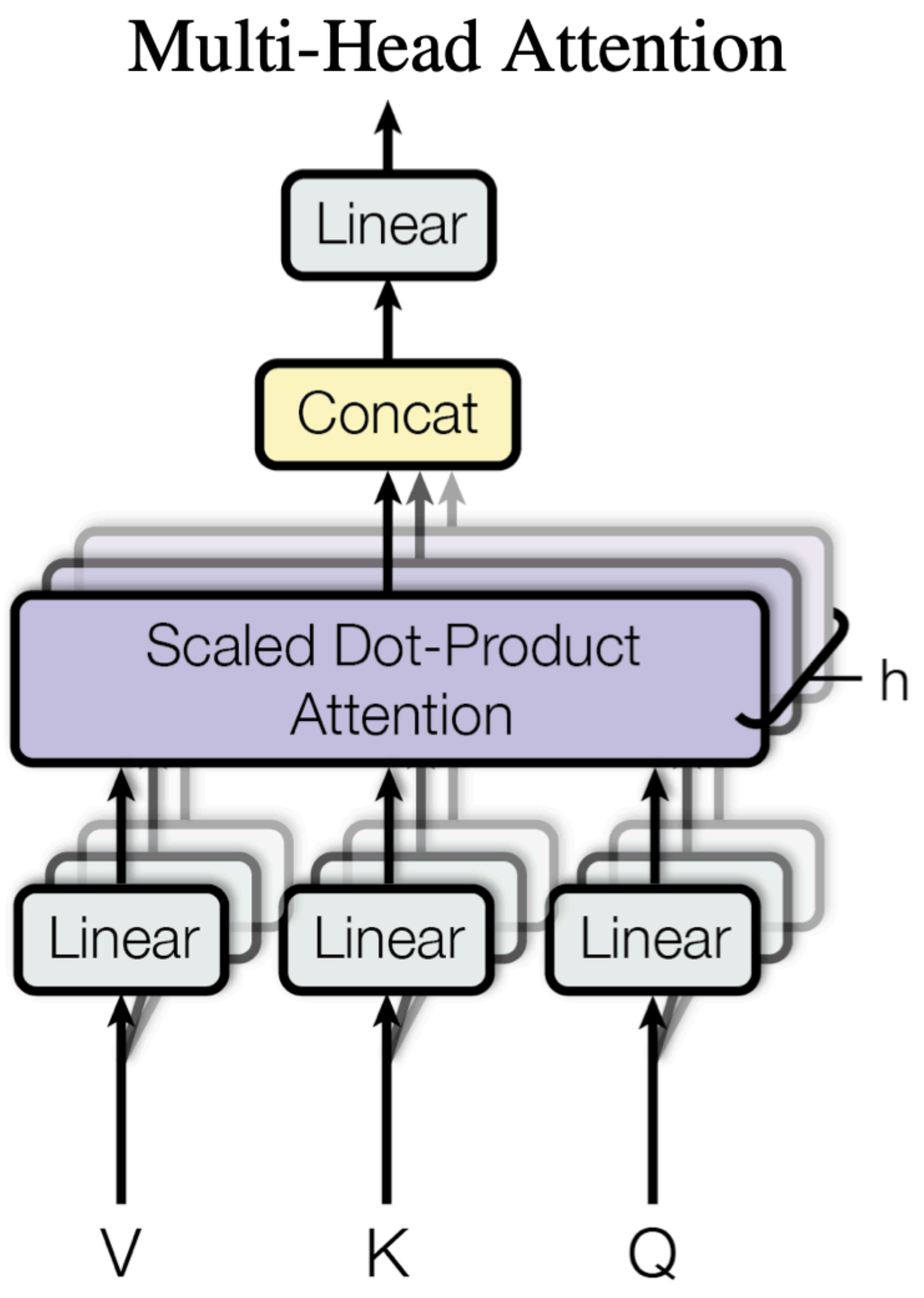
Self-Attention



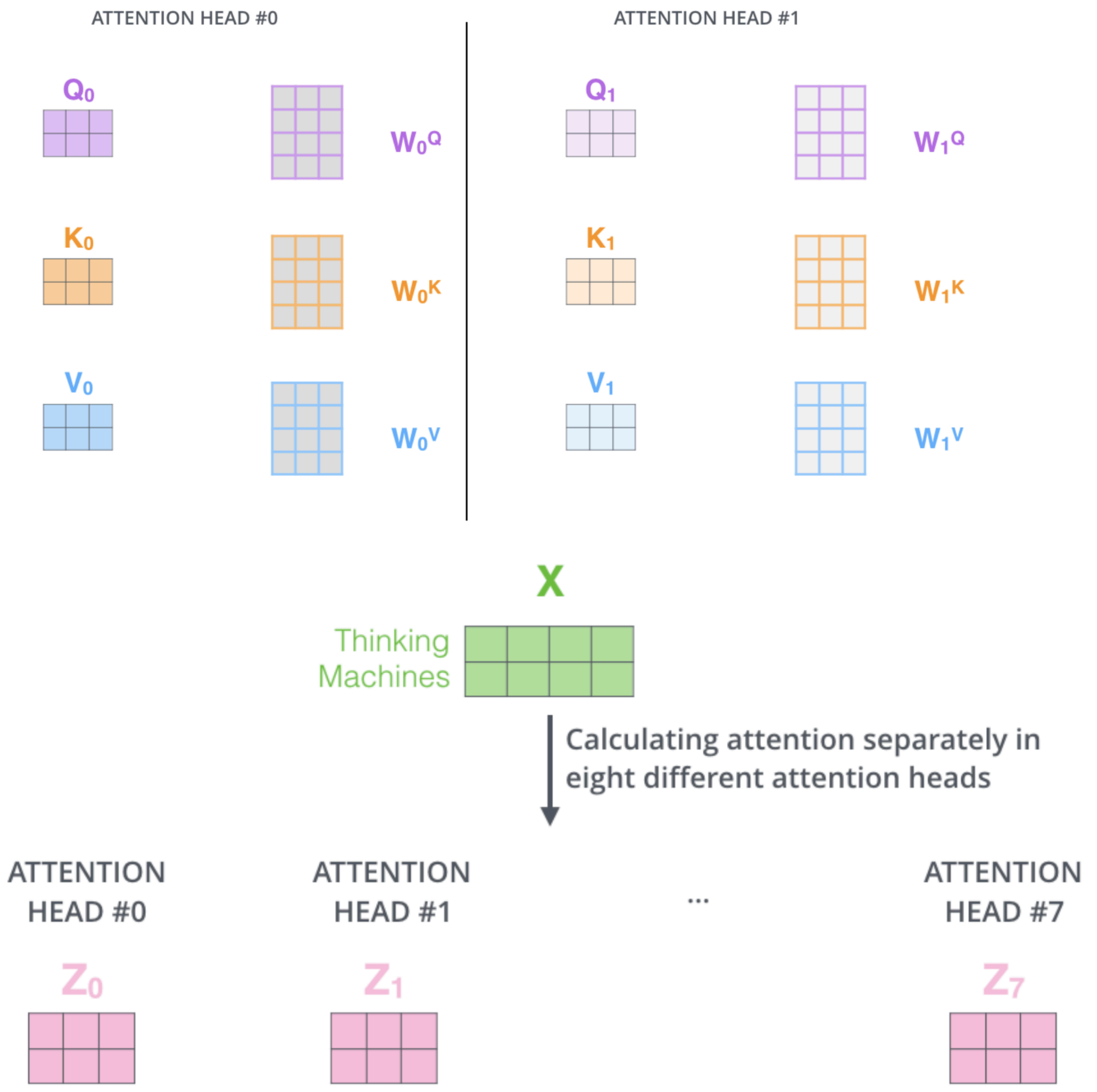
Self-Attention



Multi-Head Attention

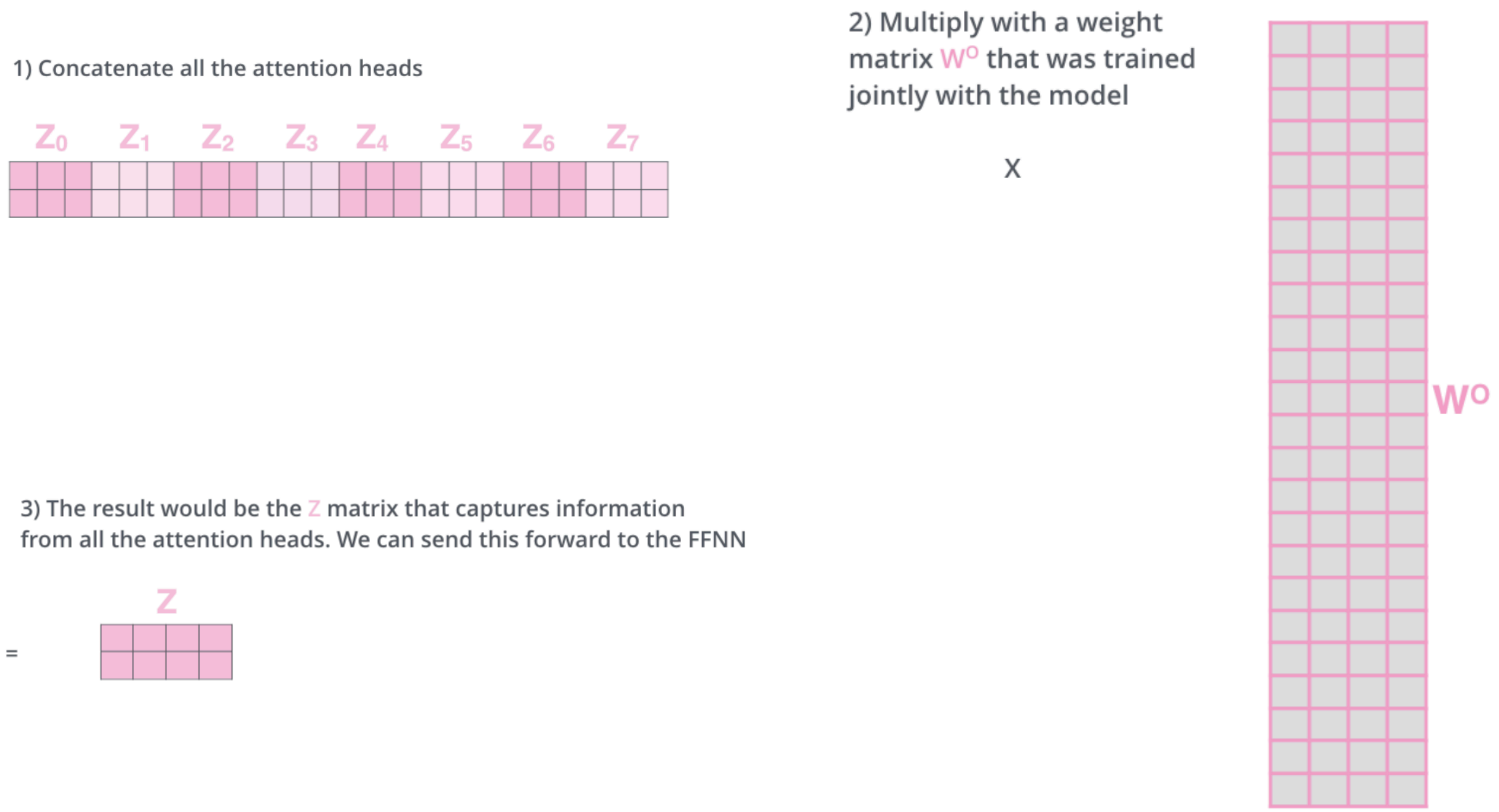


Multi-Head Self-Attention



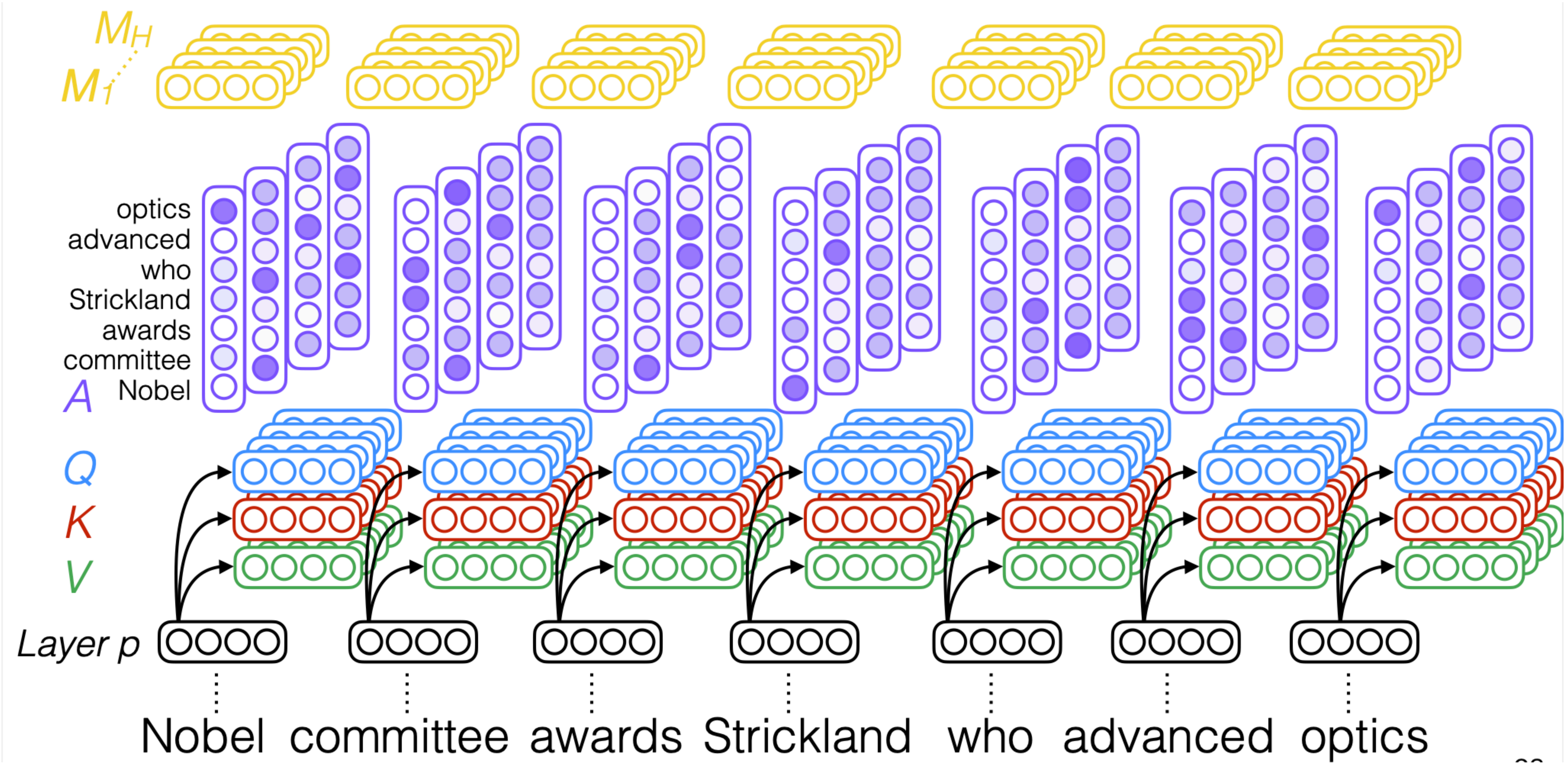
Jay Alammar. The Illustrated Transformer.

Multi-Head Self-Attention



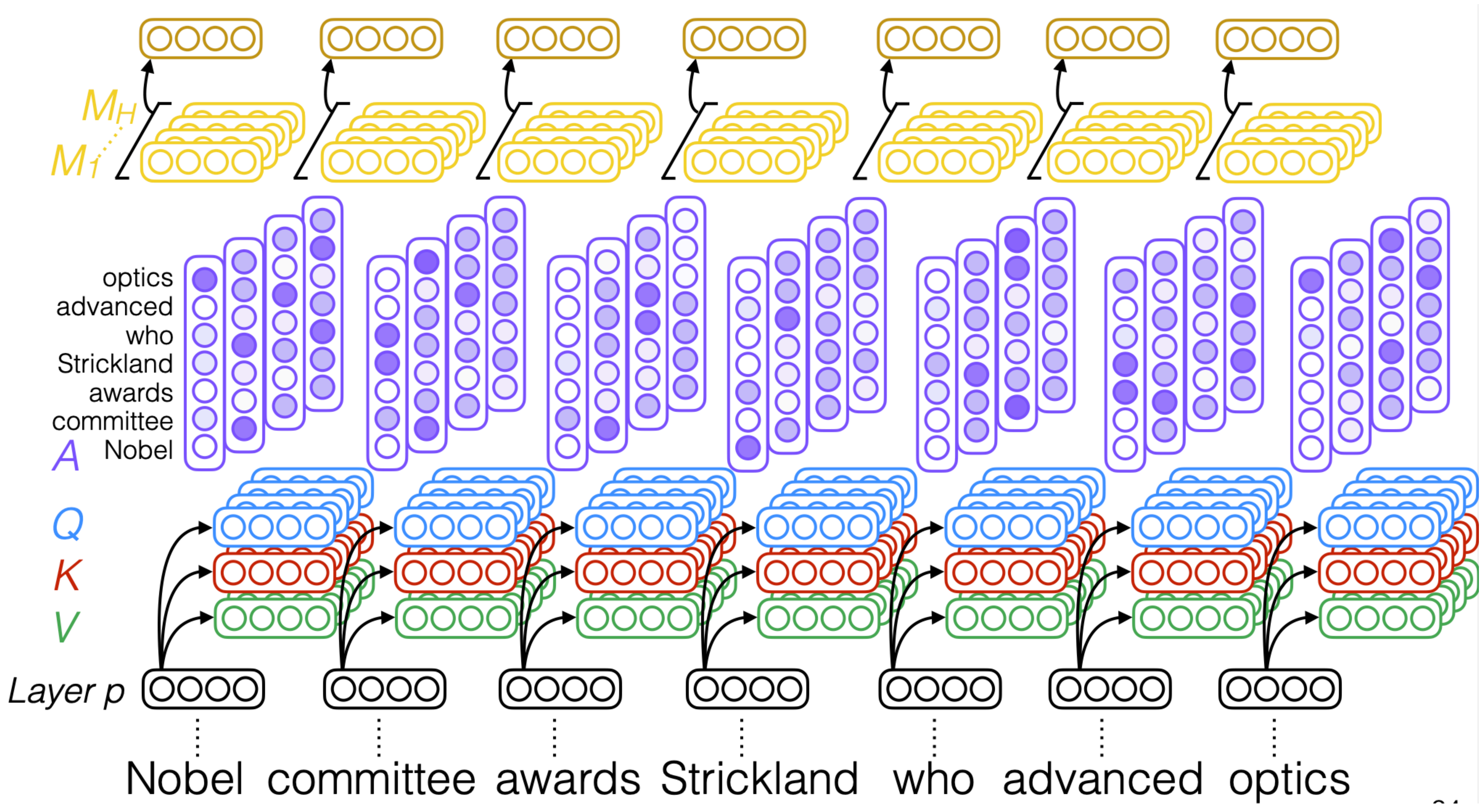
Jay Alammar. The Illustrated Transformer.

Multi-head Self-Attention

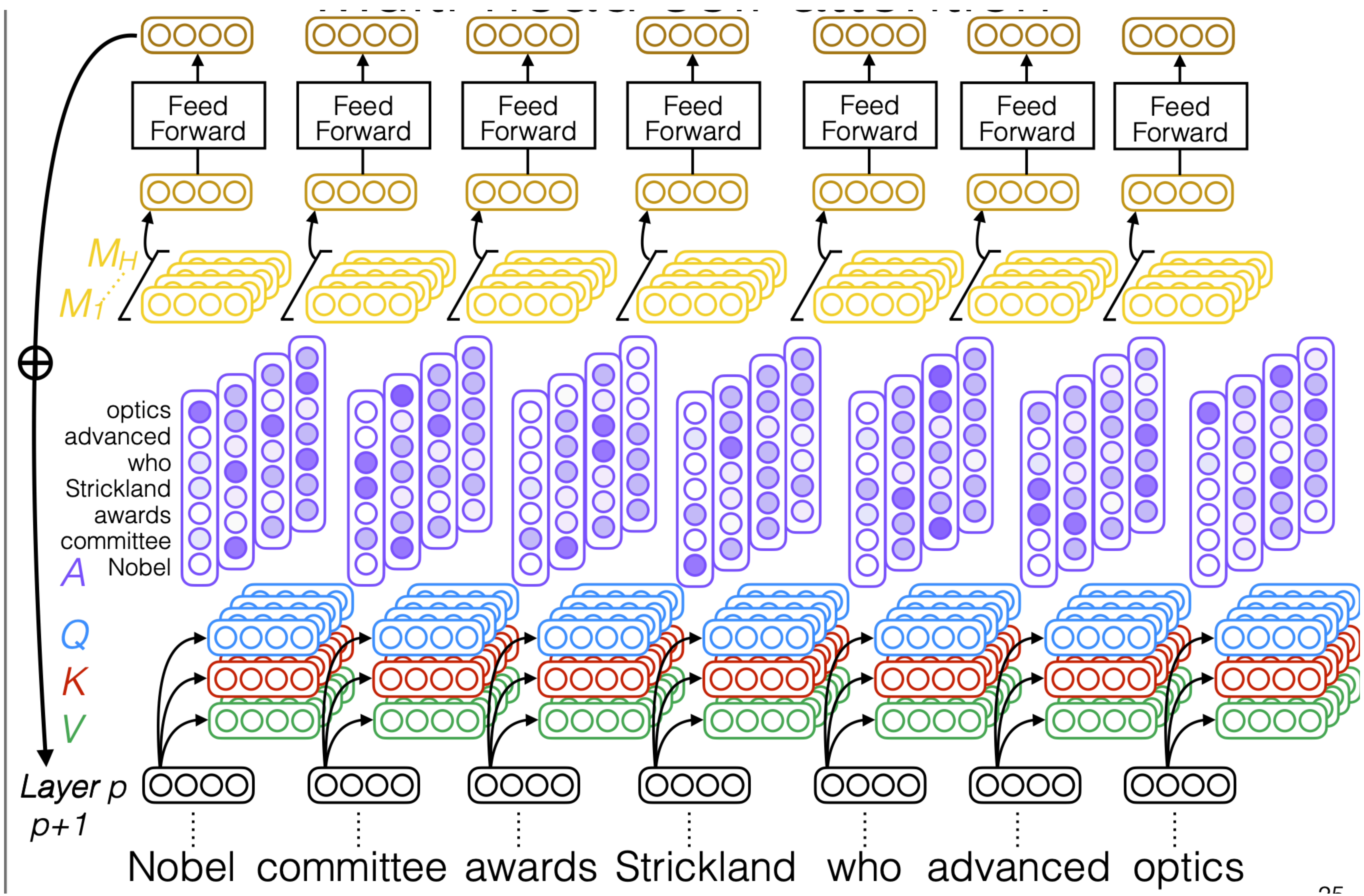


Multi-head Self-Attention

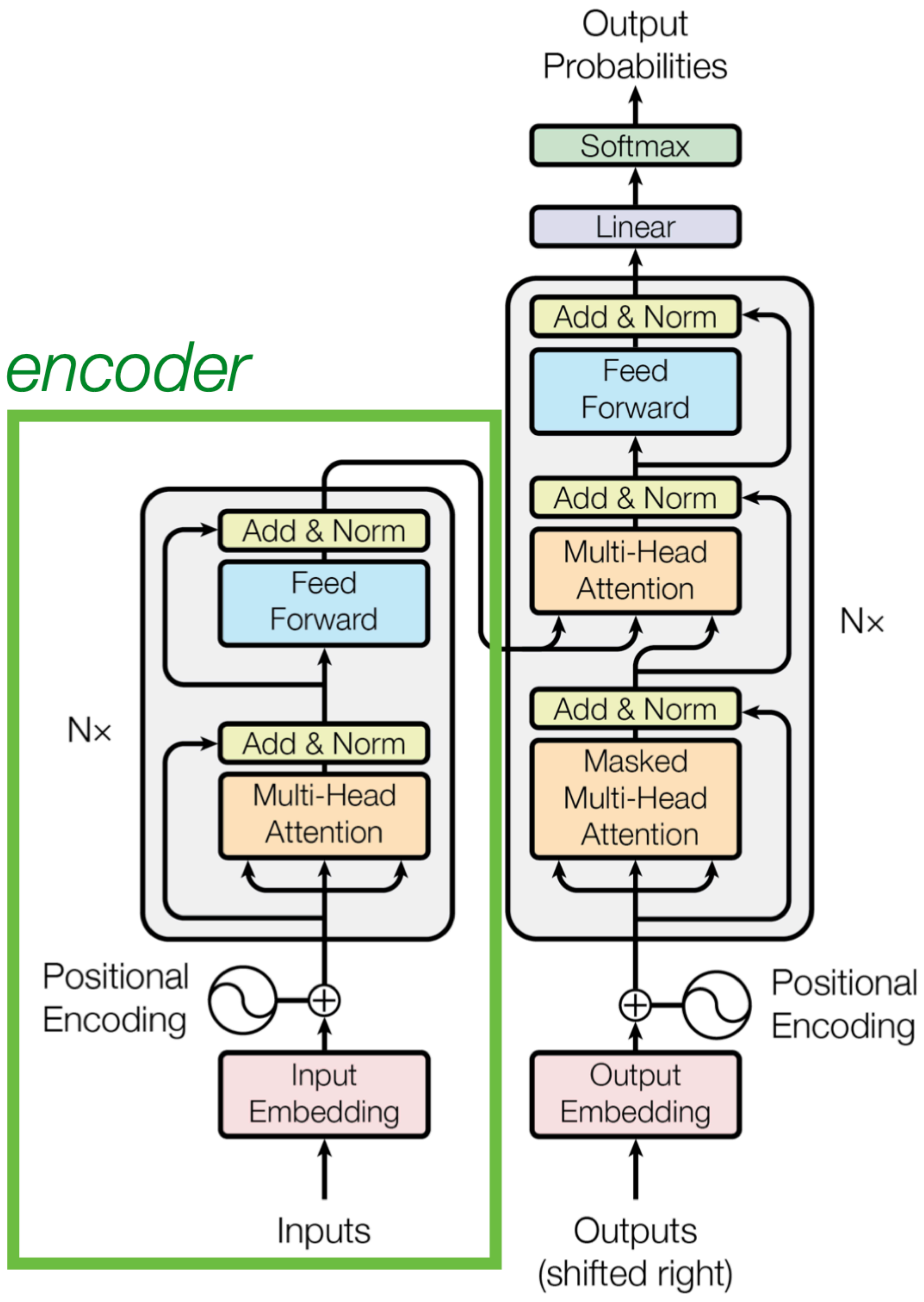
Concat and output projection



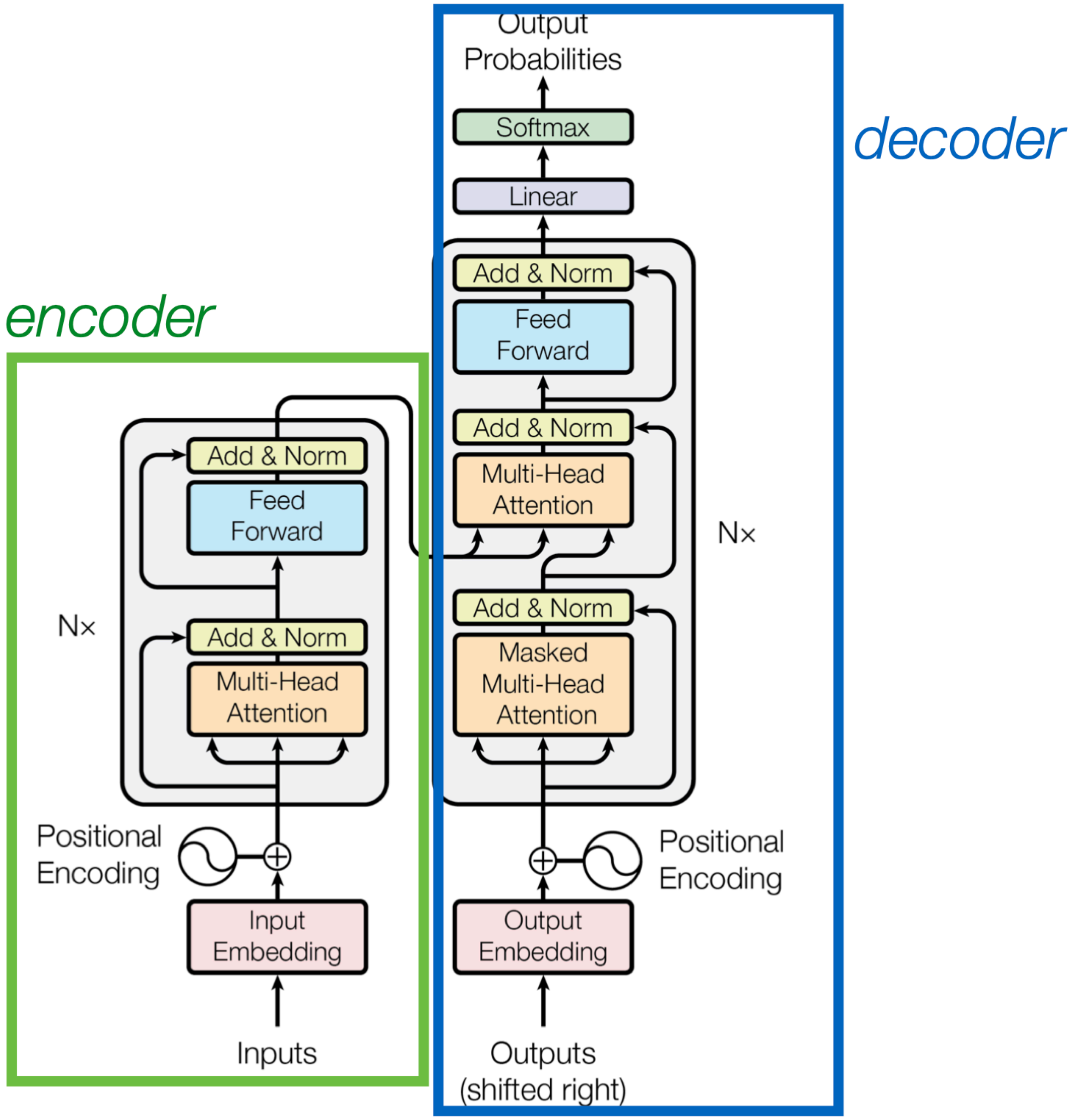
Multi-head Self-Attention + FFN



Transformer Encoder



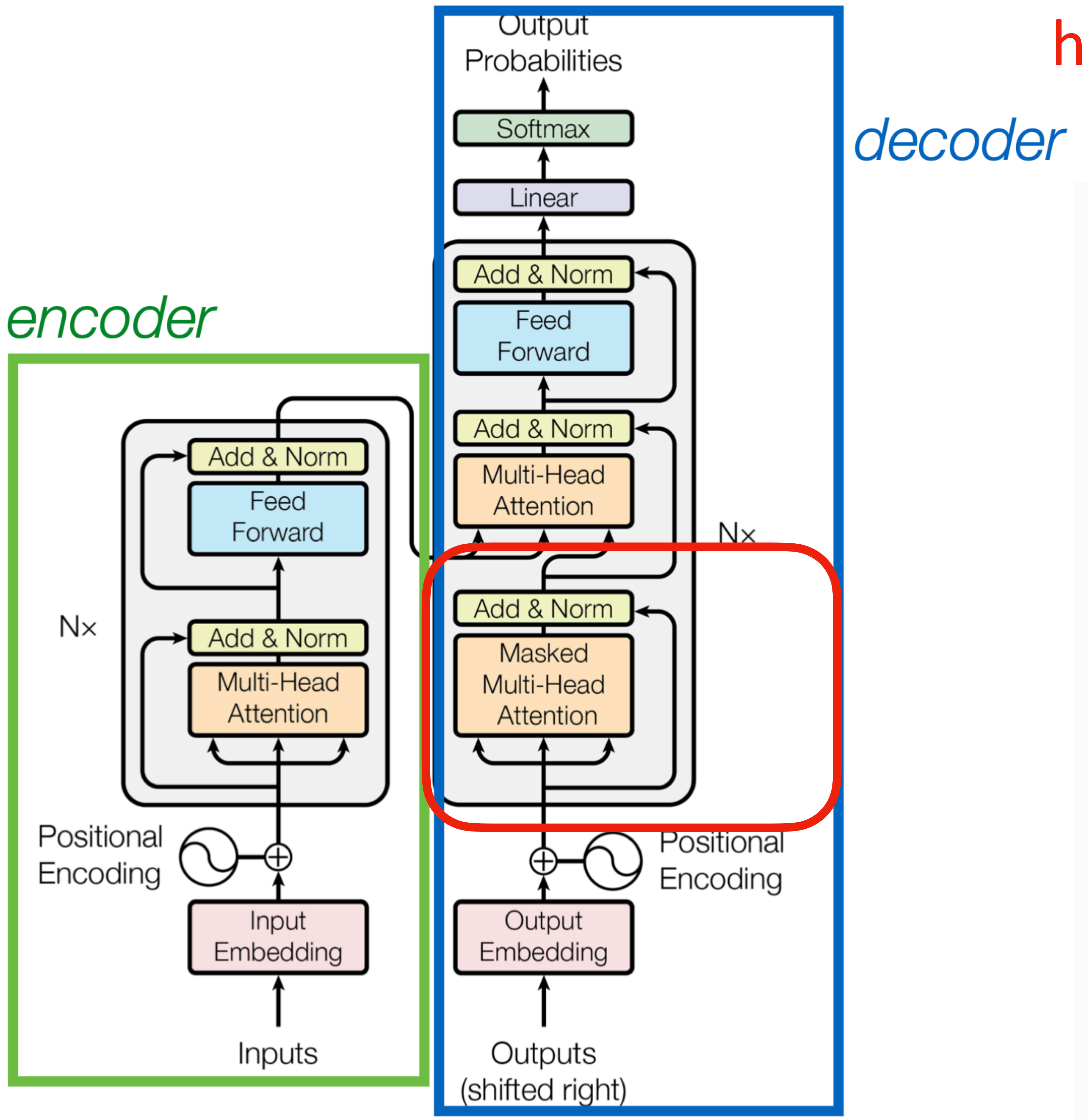
Currently we only cover the encoder side

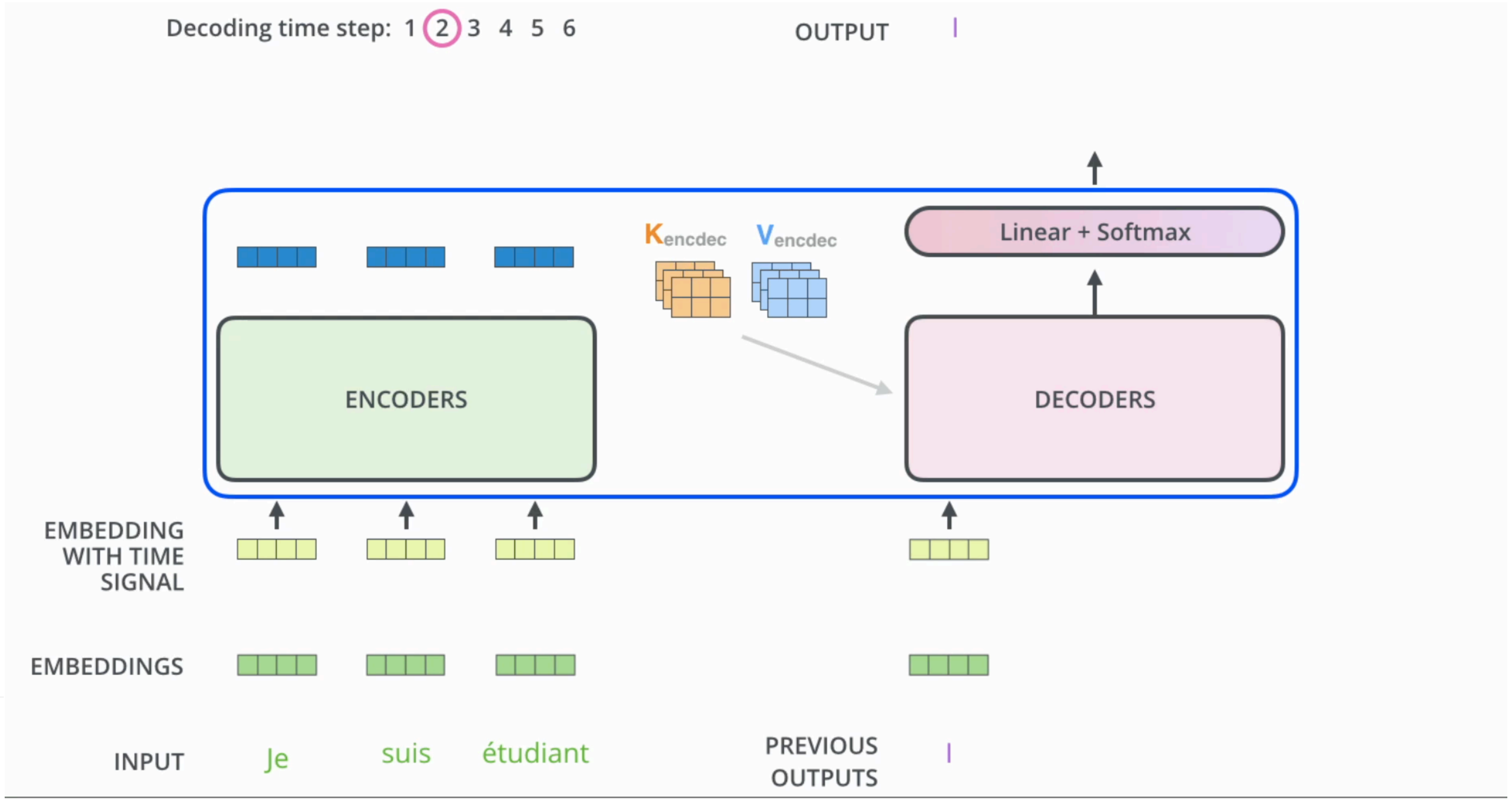


This encoder-decoder arch is originally proposed as a seq2seq arch, for classification tasks, often only encoder is used. And language models often only have a decoder

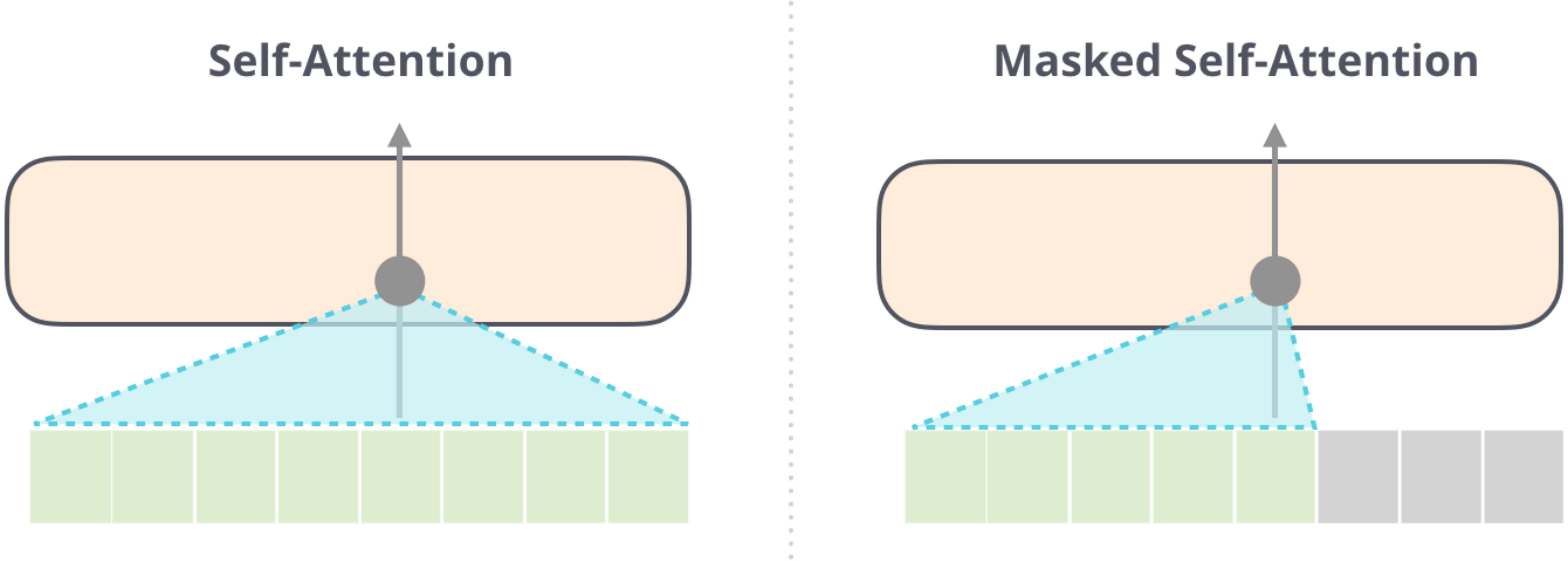
Masked Attention

Typical attention attends to the entire sequence, while masked attention only attends to the ones on the left because future words have not been generated

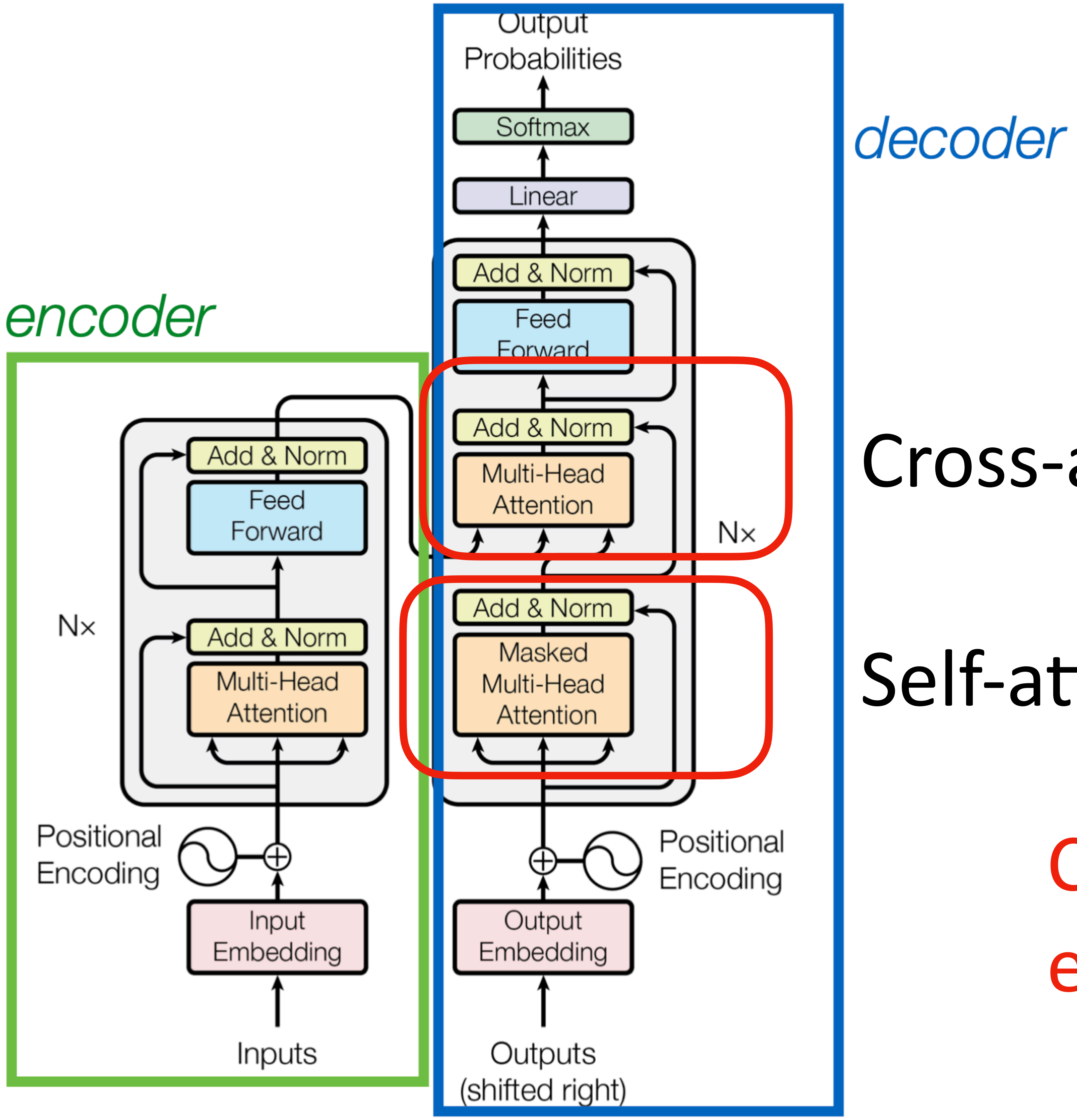




Masked Attention

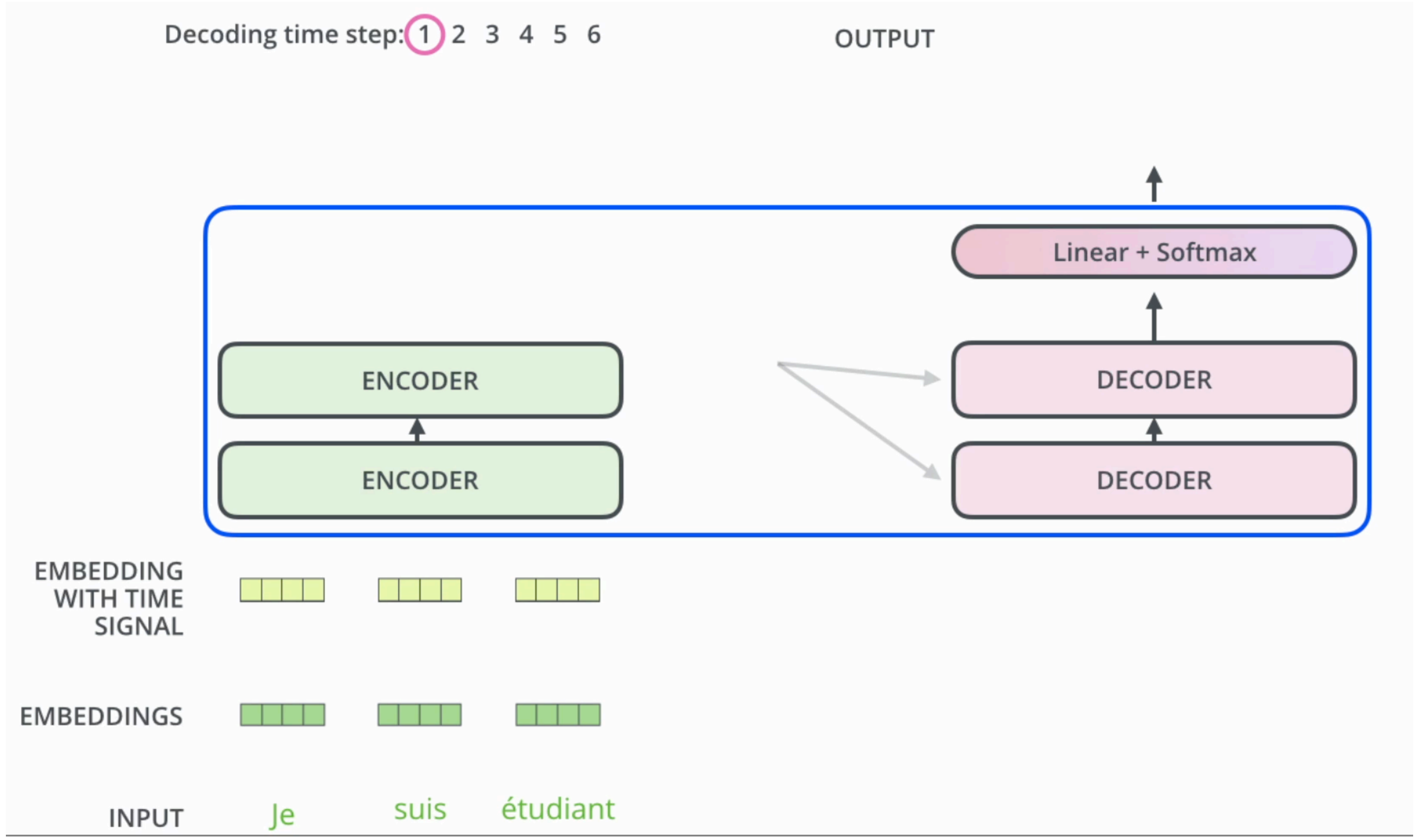


Transformer Decoder in Seq2Seq



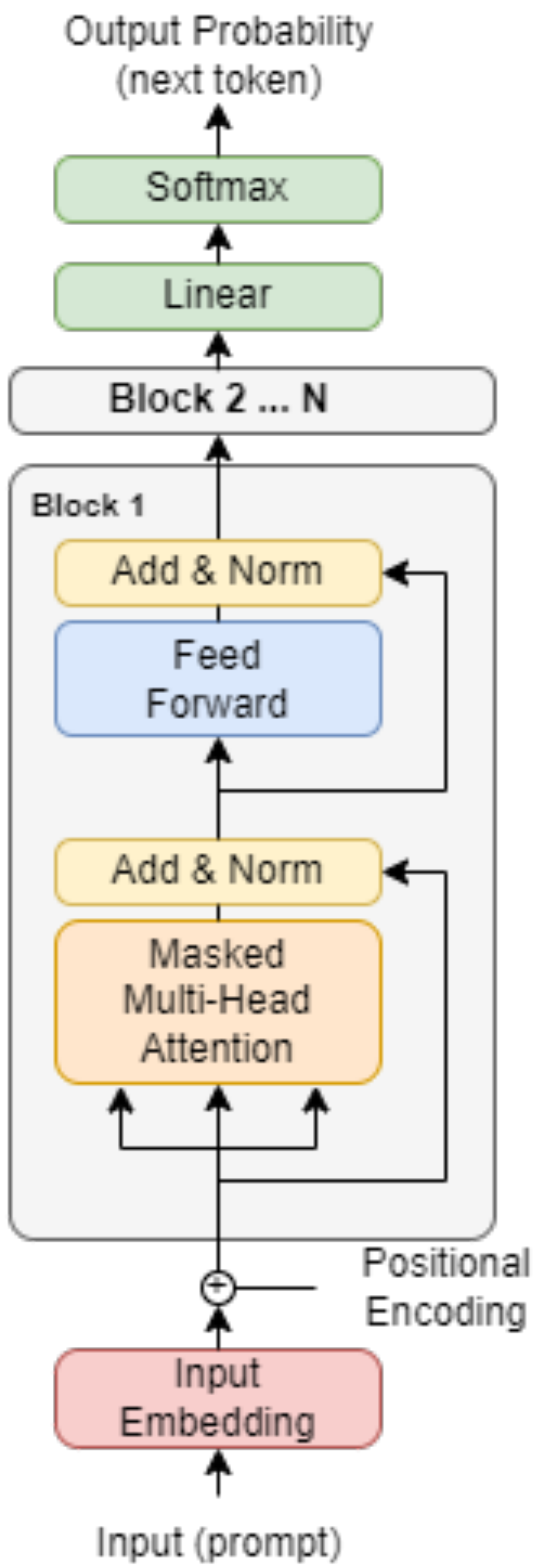
Cross-attention

Self-attention

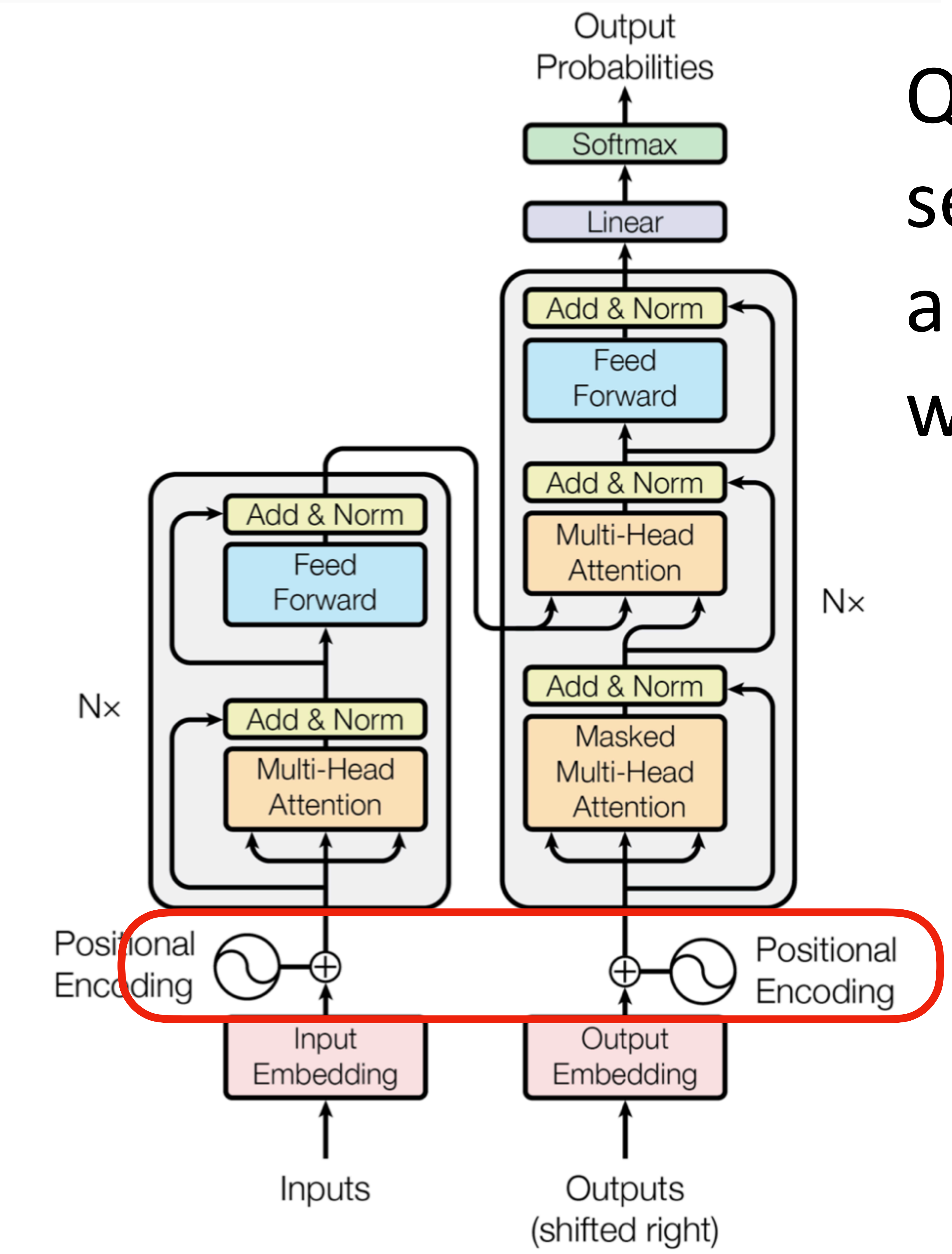


Cross-attention uses the output of encoder as input

Transformer Language Model (e.g., ChatGPT)



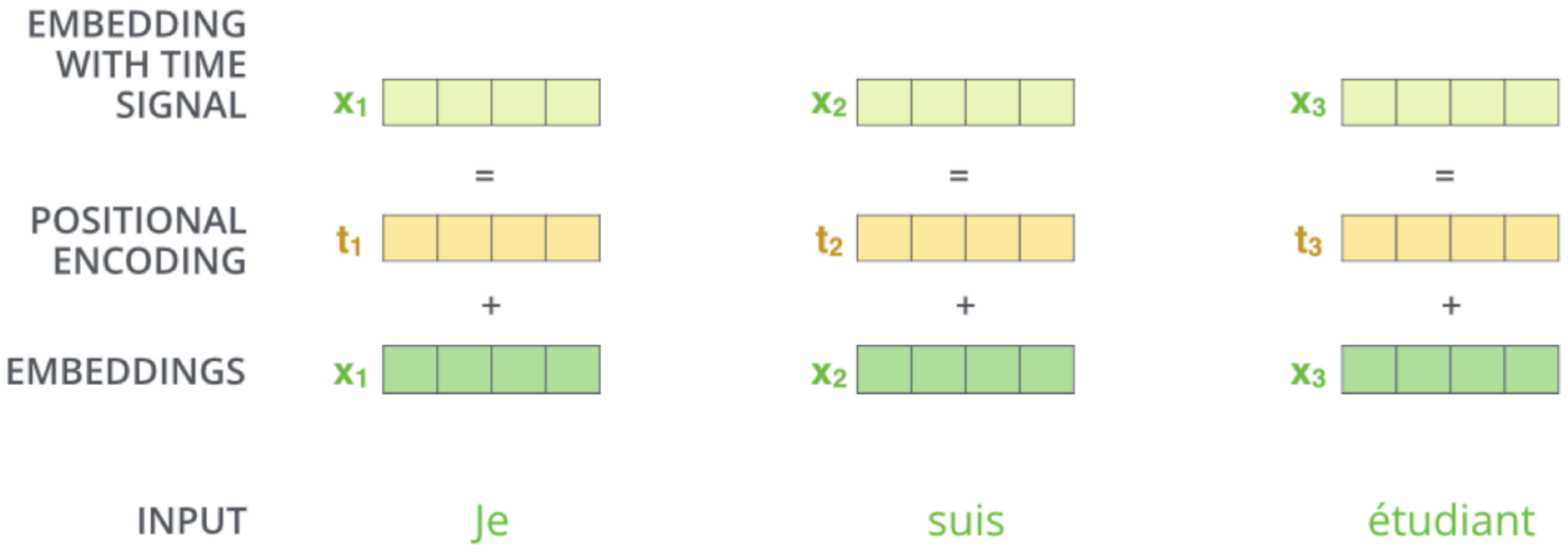
Position Embeddings



Question: If we shuffle the order of words in the sequence, will that change the attention output and feed forward output of the corresponding word?

Position embeddings are added to each word embedding, otherwise our model is unaware of the position of a word

Positional Encoding



Transformer Positional Encoding

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

Positional encoding is a 512d vector
i = a particular dimension of this vector
pos = dimension of the word
d_model = 512

Complexity

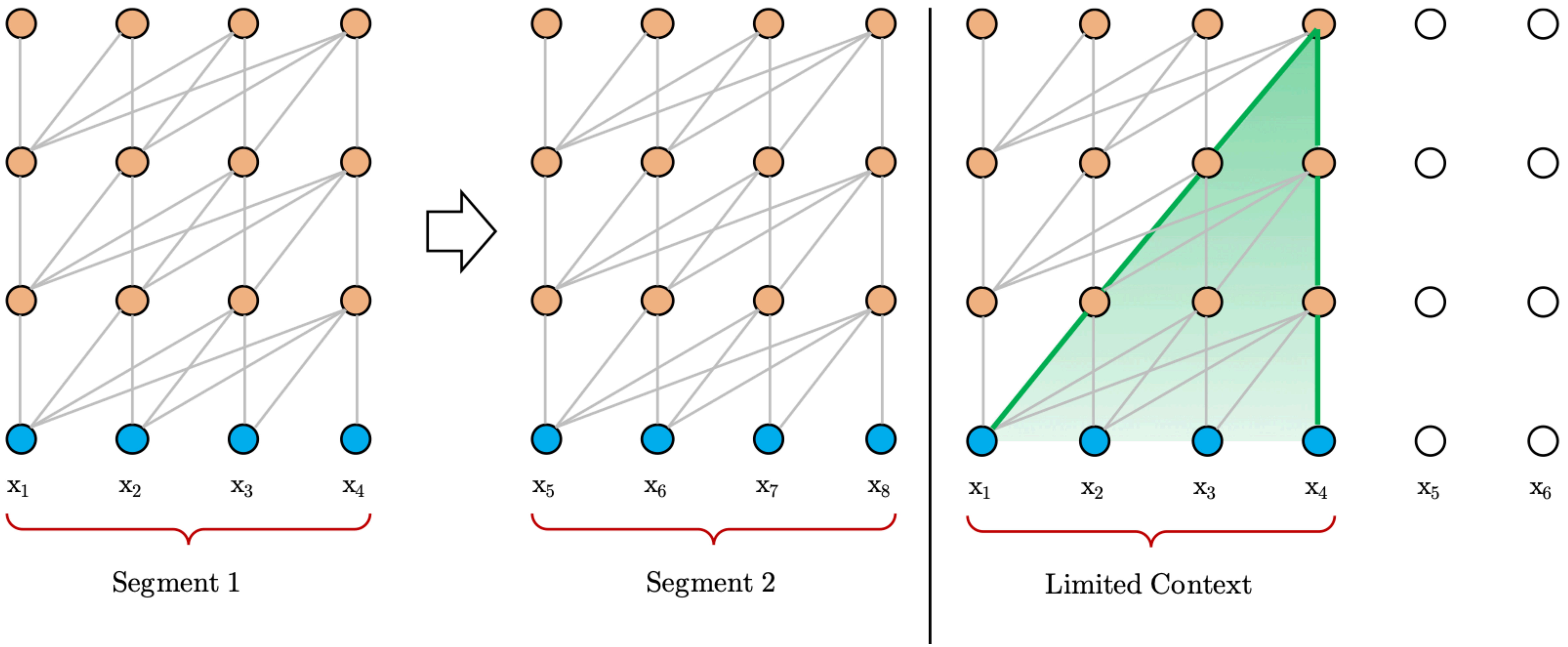
Layer Type	Complexity per Layer	Sequential Operations
Self-Attention	$O(n^2 \cdot d)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$

n is sequence length, d is embedding dimension.

Restricted self-attention means not attending all words in the sequence, but only a restricted field

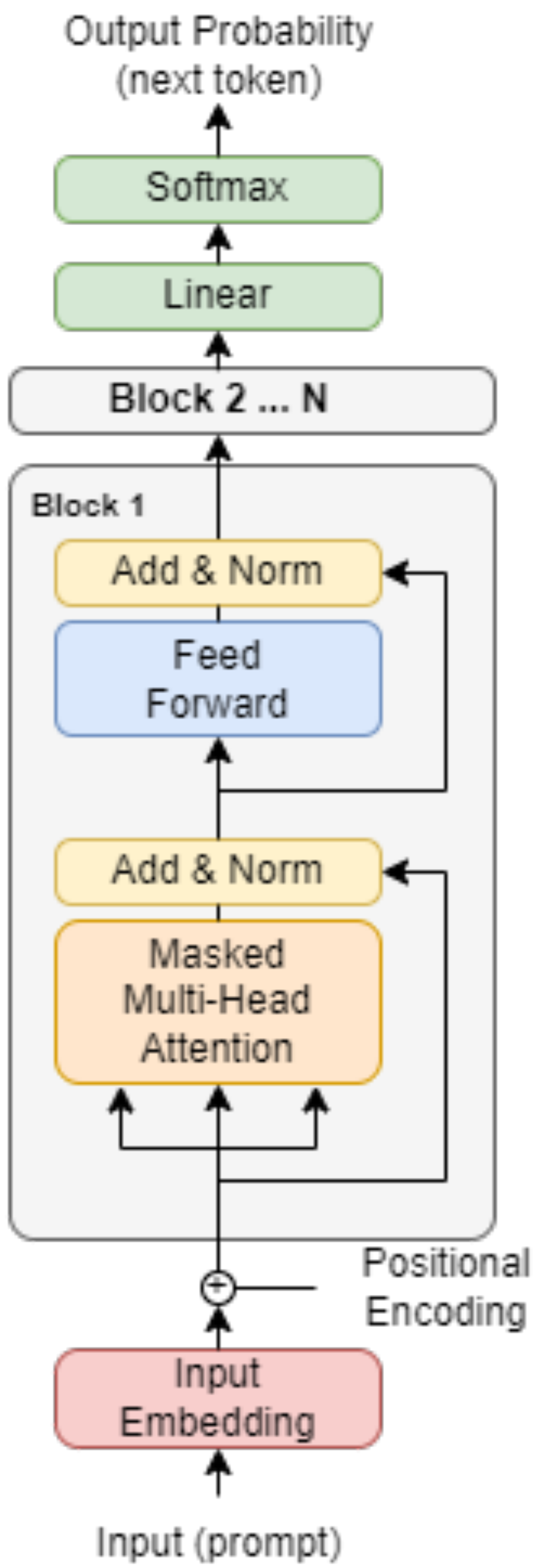
Square complexity of sequence length is a major issue for transformers to deal with long sequence

Language Model Training with Limited Context



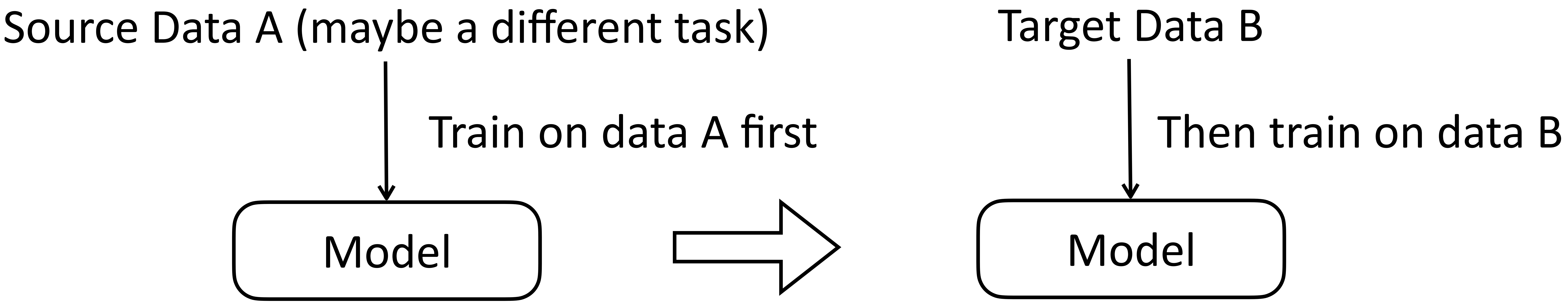
Dai et al. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. 2019.

Transformer Language Model (e.g., ChatGPT)



Language Model Pretraining

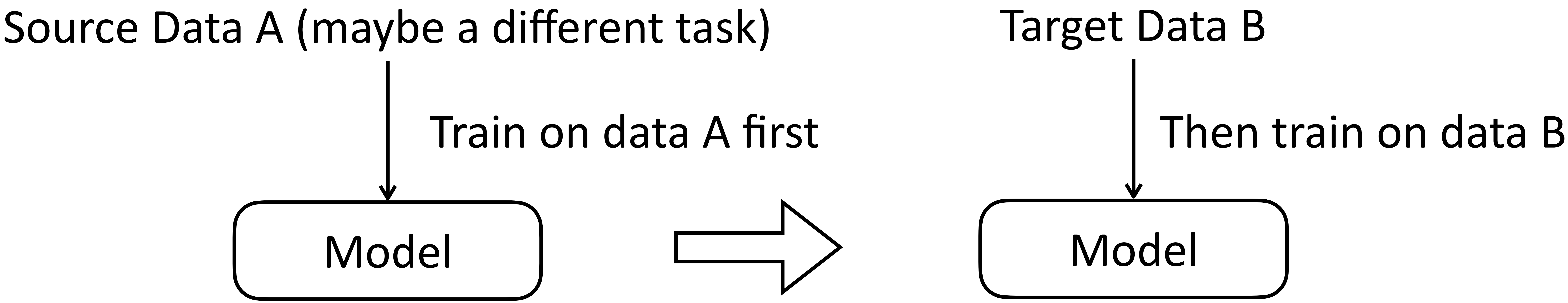
Pretraining



Classically, this is transfer Learning

It is now called pretraining because of the scale of A

Pretraining



For supervised training, data A is often limited

How can we find large-scale data A to train?

BERT

Mask language modeling

mouse

Transformer

<start> The [mask] ate the cheese

Self-supervised Learning

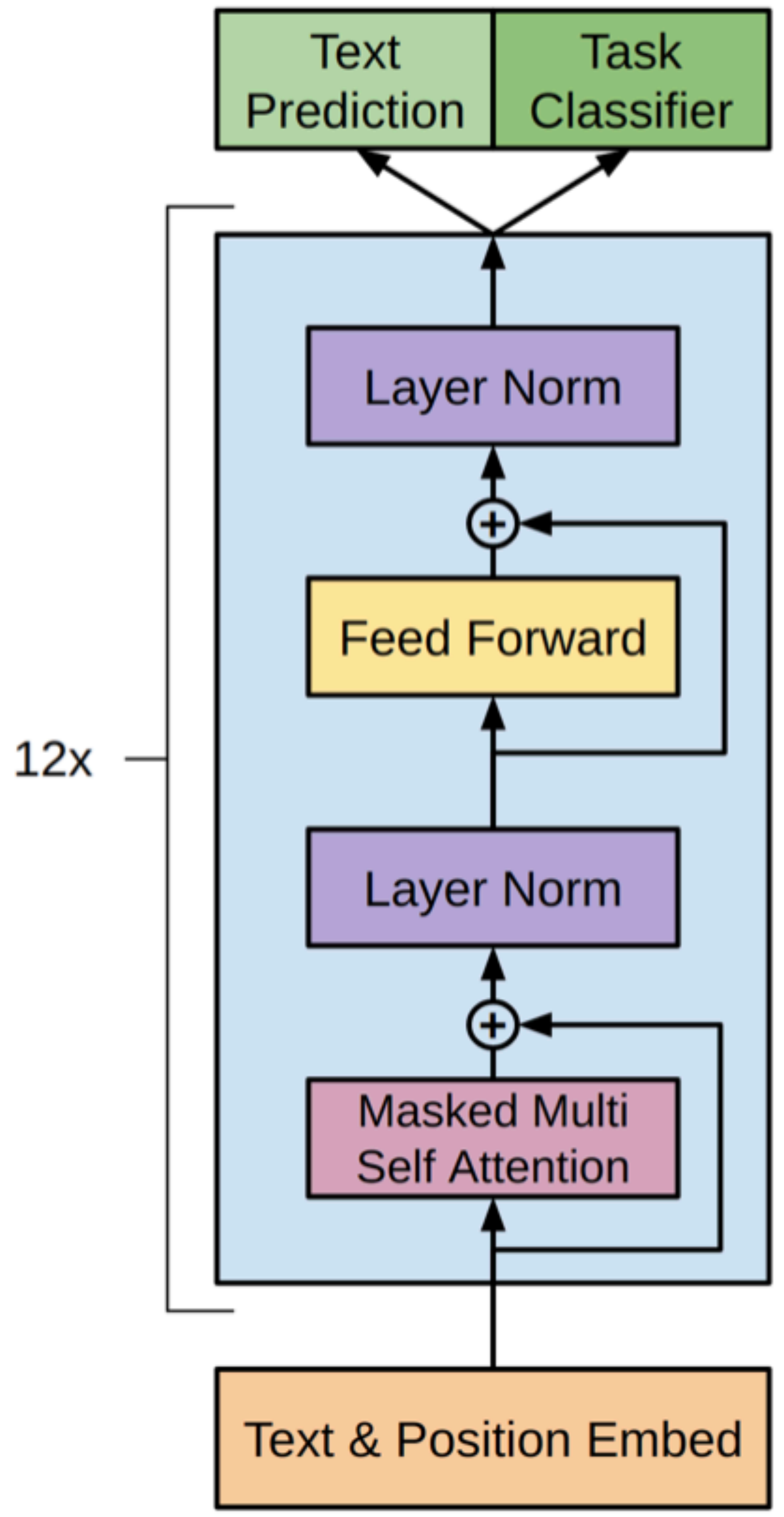
Construct a synthetic task from raw text only

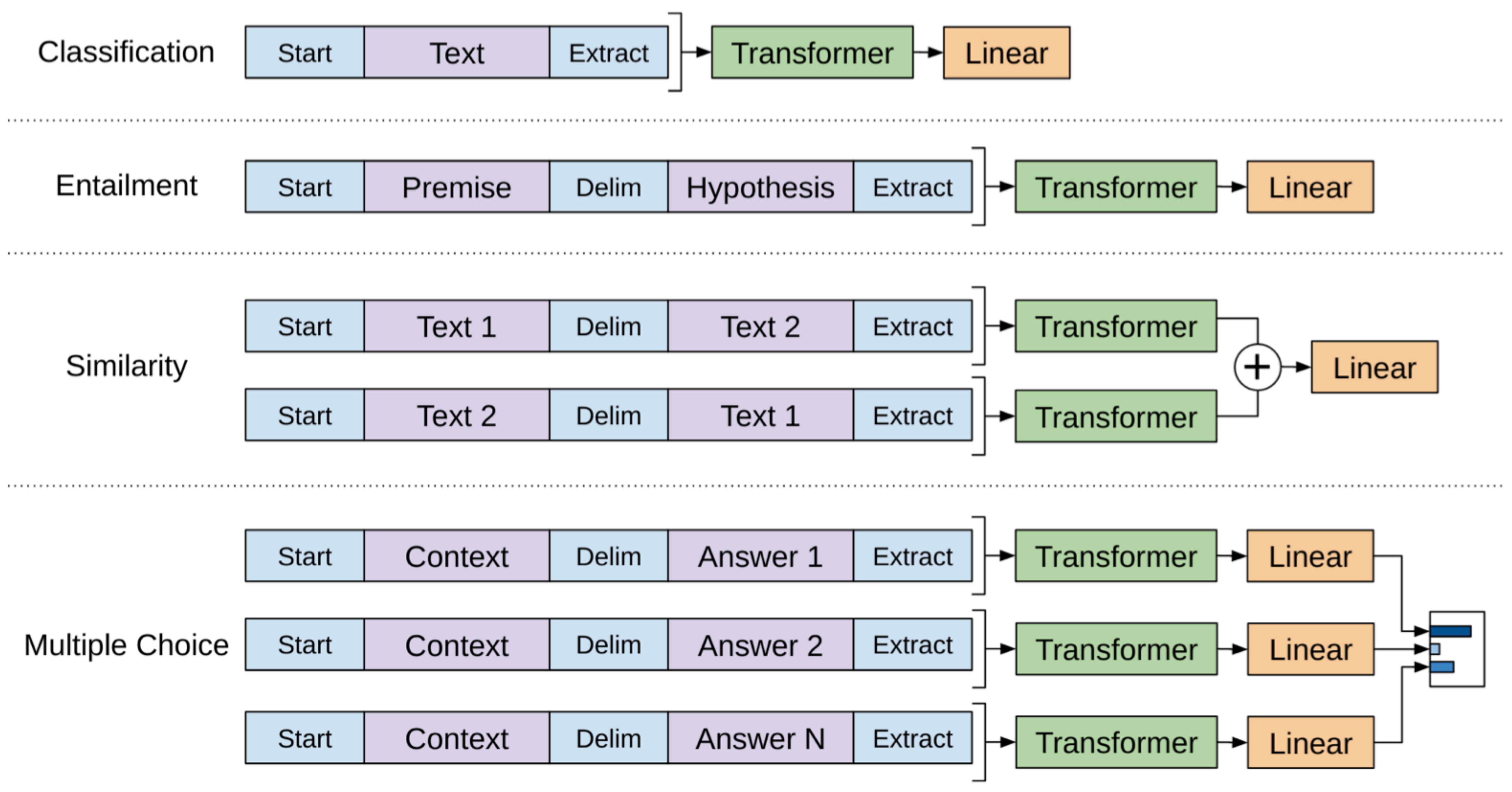
Can be made very large-scale

Is Bert a language model? Is it a generative model?

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL 2019.

Generative Pre-Training (GPT)





Radford et al. Improving Language Understanding by Generative Pre-Training. 2018

Thank You!