Transformers

CS 4804: Introduction to AI

Fall 2025

https://tuvllms.github.io/ai-fall-2025/

Tu Vu

VIRGINIA TECH.

Logistics

- Homework 0 (due September 16th)
  - accuracy
- Quiz 0 (due tomorrow)
  - genuine attempt
- Final project group information (tomorrow)

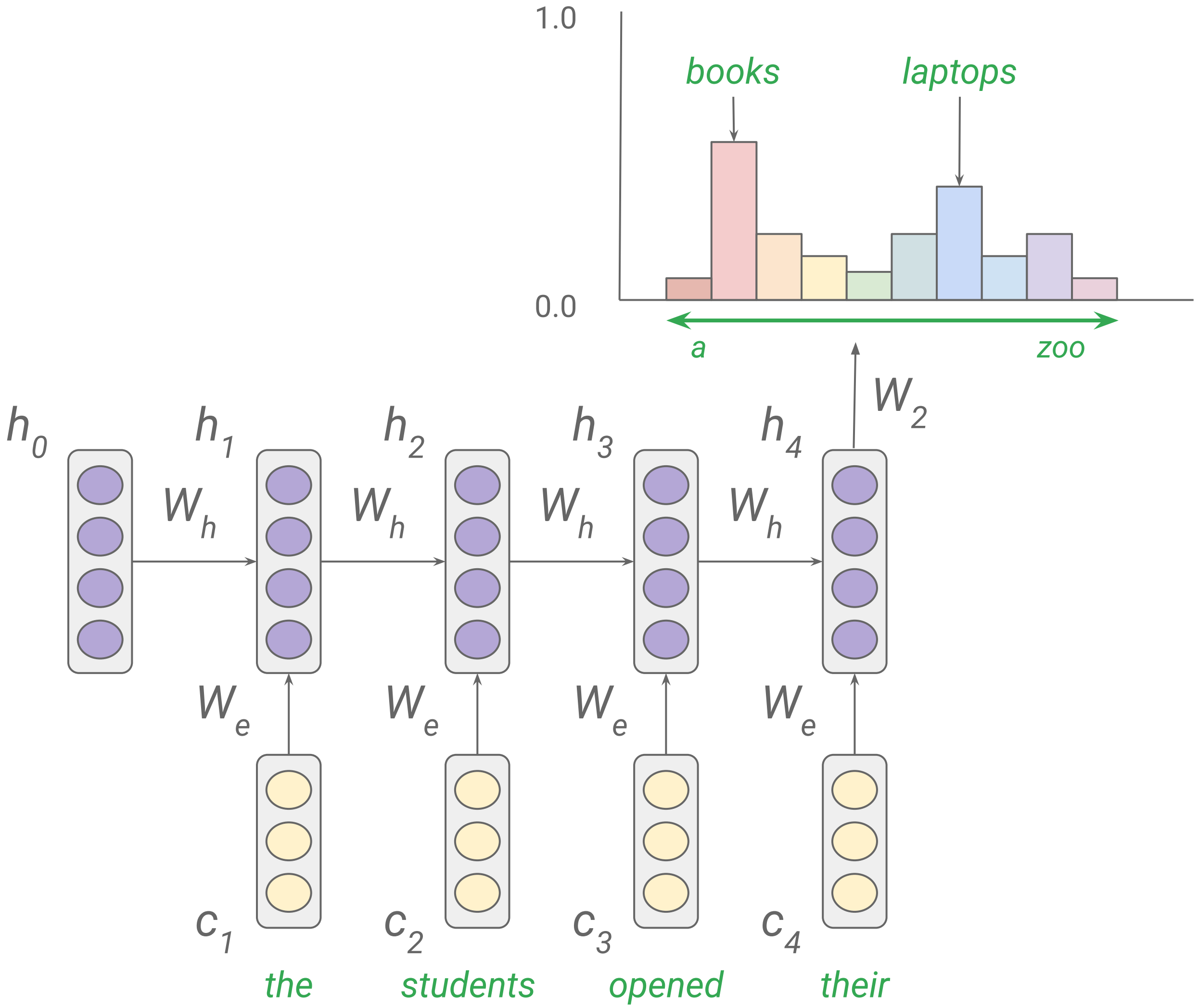
Recurrent neural networks (RNNs)

hidden states

$$h^{(t)} = f(W_h h^{(t-1)} + W_e c^t)$$

output distribution

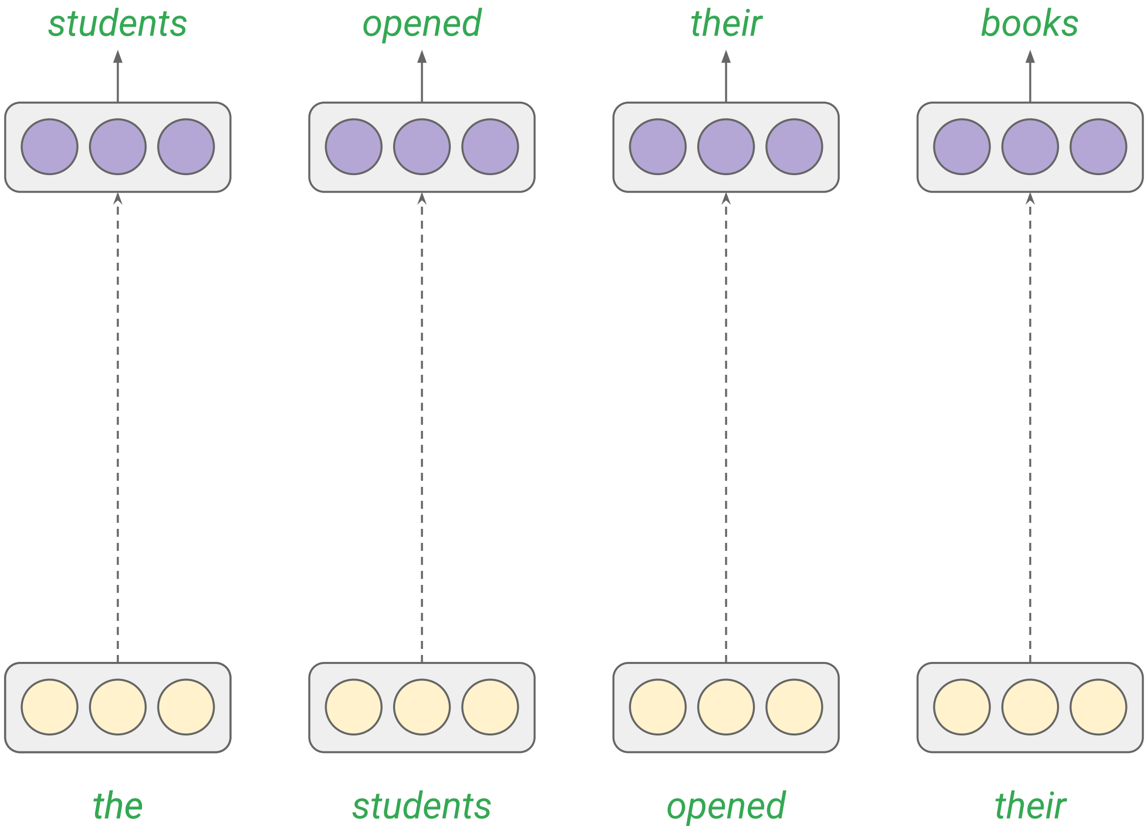
$$\hat{y} = softmax(W_2 h^{(n-1)})$$



Problems with RNNs

- Bottleneck representation issue
- Lack of parallelism

Seq2Seq



Transformers

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

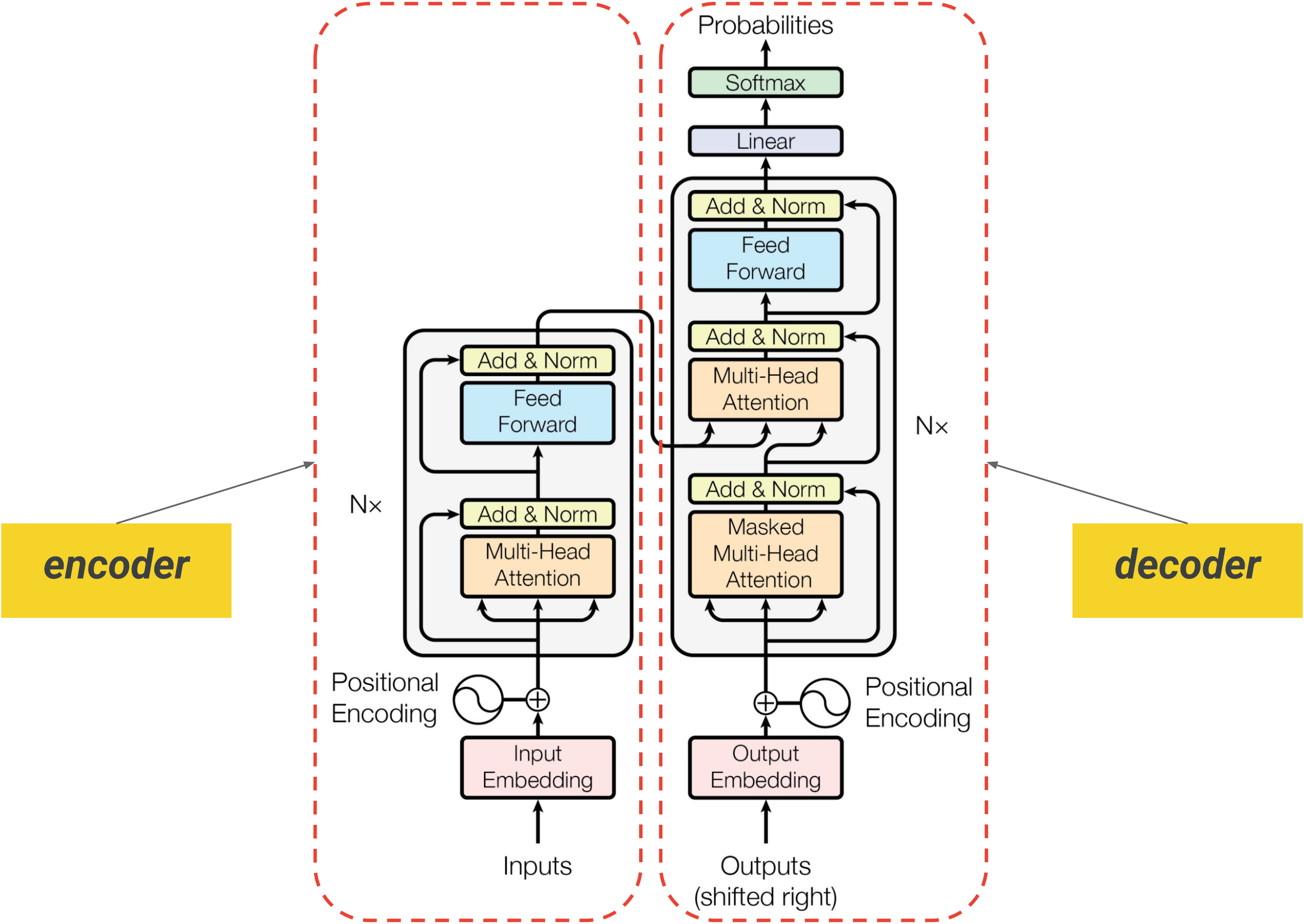
Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

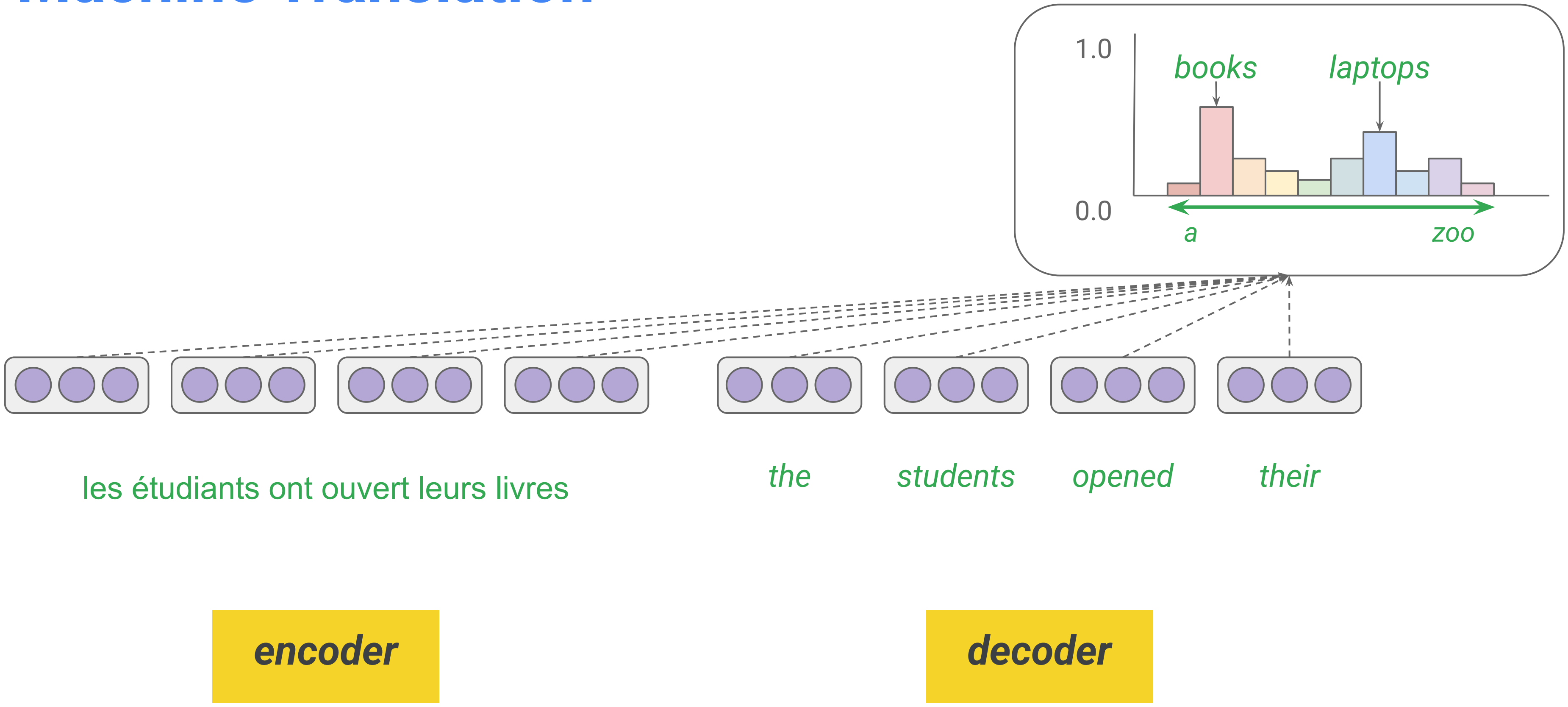
Transformers

- Before 2017
  - Recurrent neural networks (RNNs)
    - LSTM (Long Short-Term Memory)
  - Convolutional neural networks (CNNs)

- These days
  - Transformers


encoder
decoder
Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

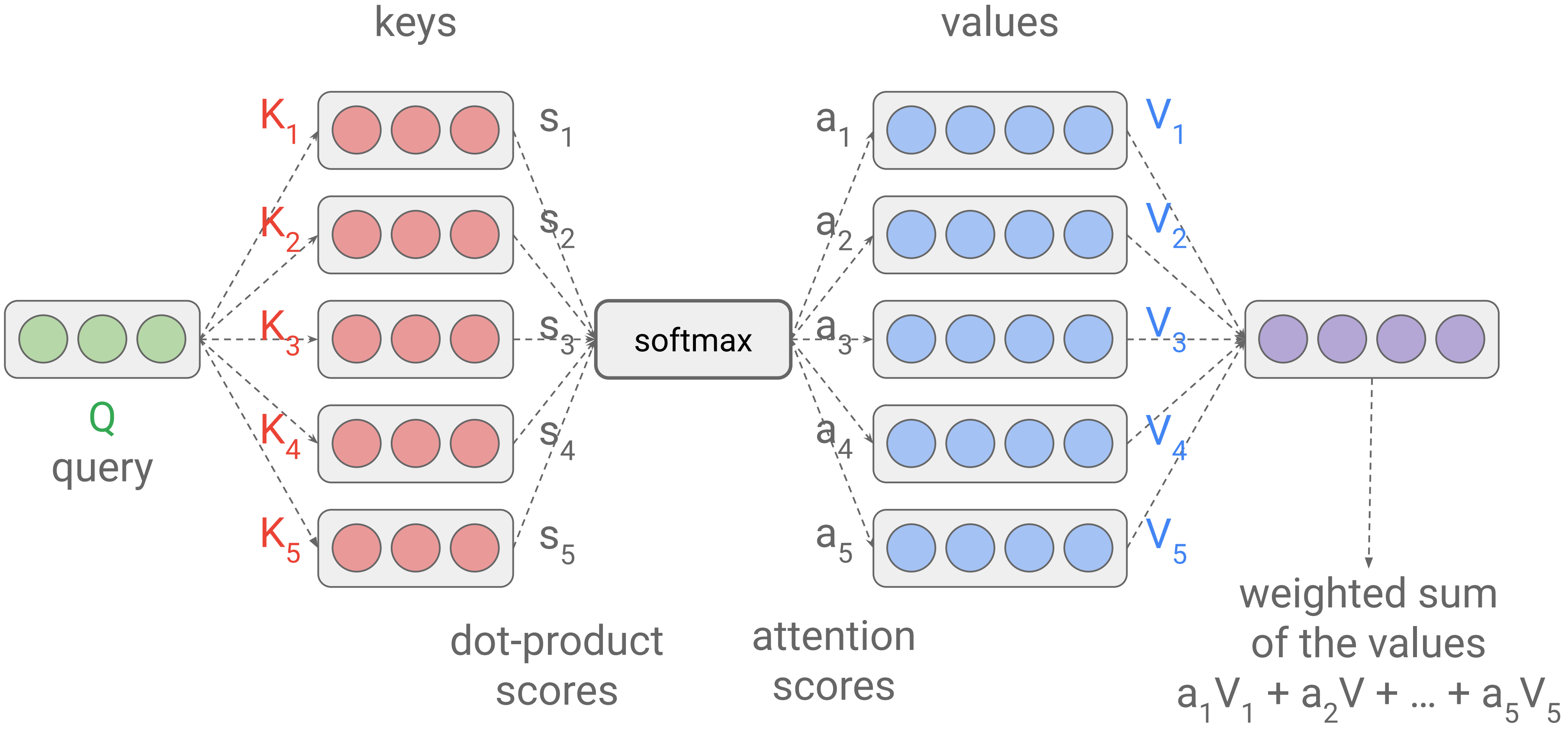
Machine Translation



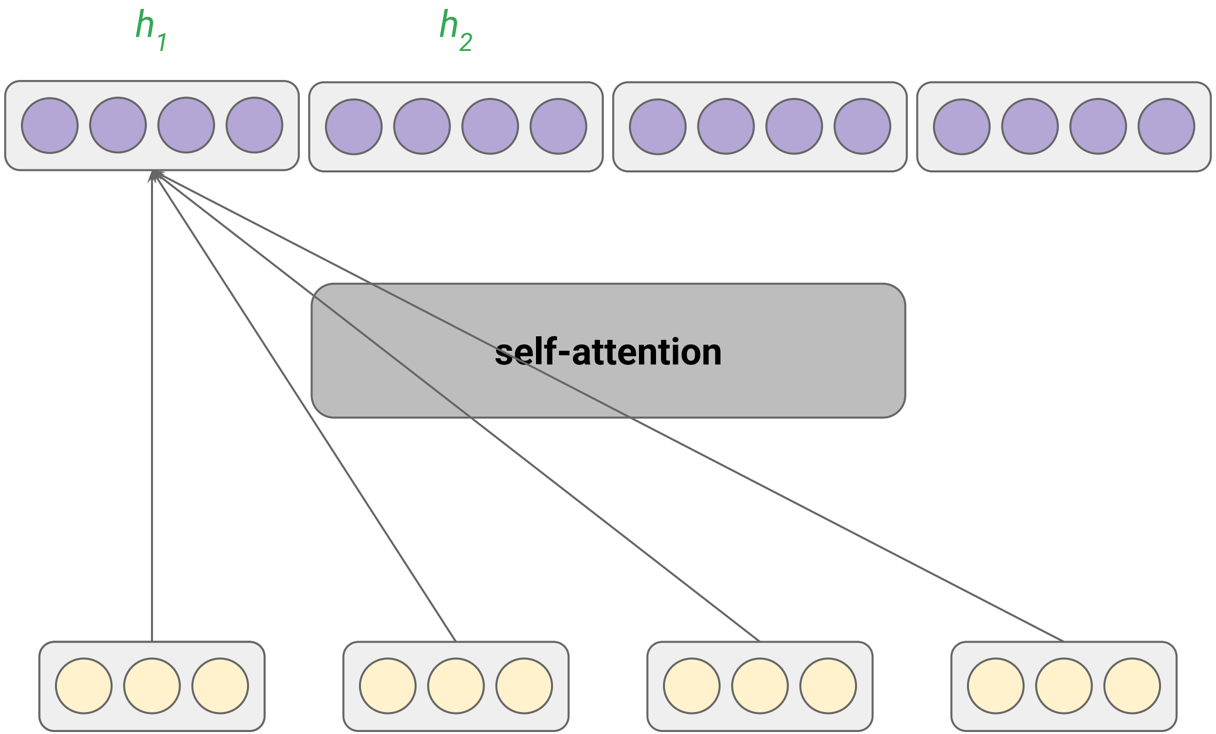
Different model architectures

- Encoder-only
  - BERT
- Encoder-decoder
  - T5
- Decoder-only
  - GPT

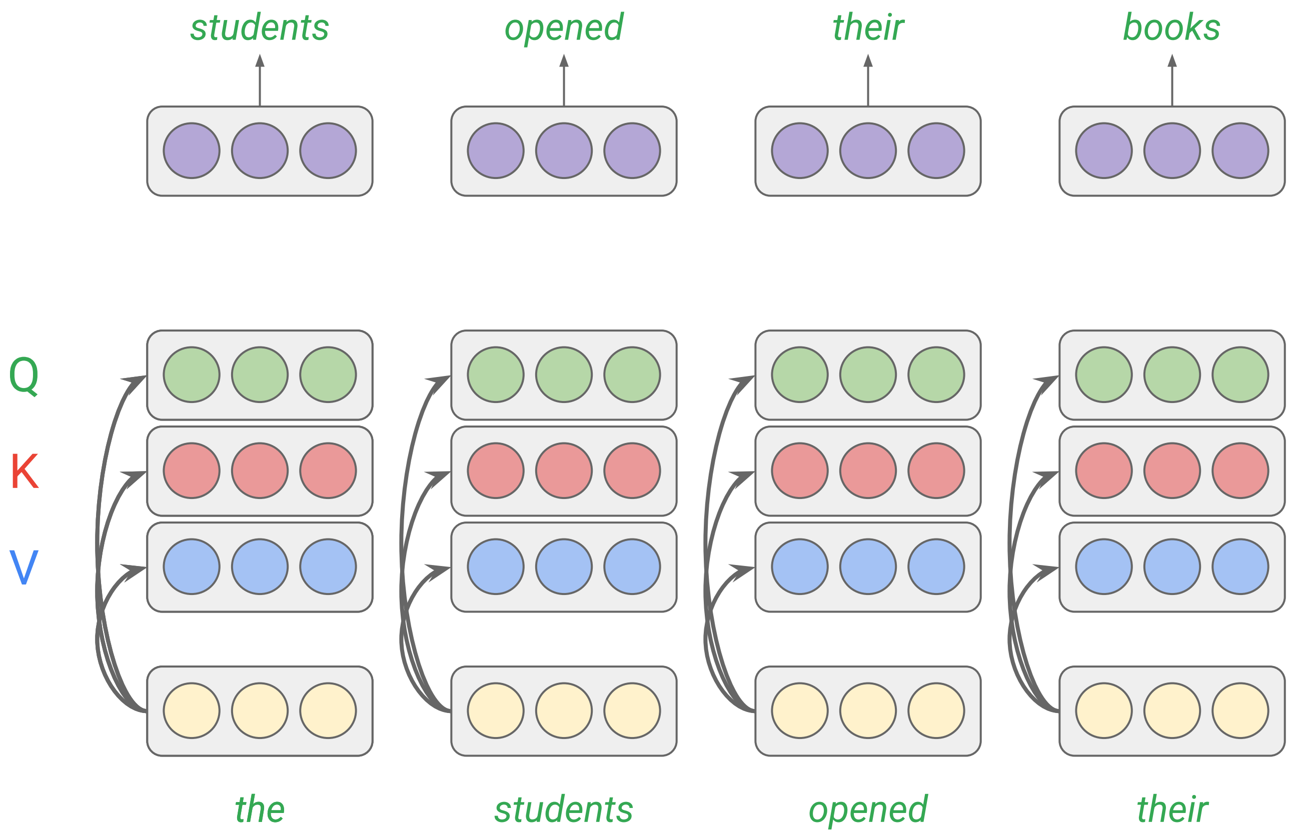
Attention mechanism



Self-attention



Self-attention (cont'd)



$$Q = X \cdot W_Q$$

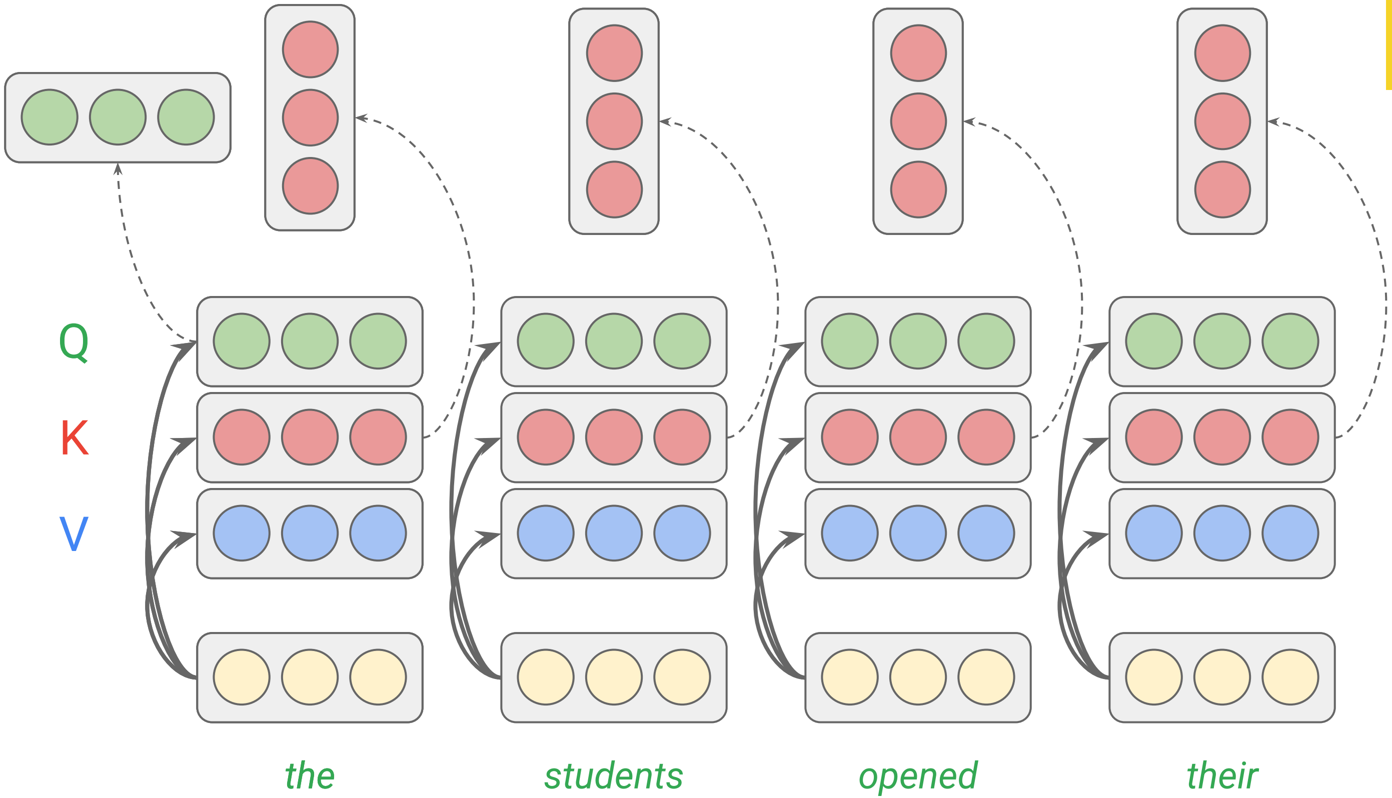
$$K = X \cdot W_K$$

$$V = X \cdot W_V$$

linear projections

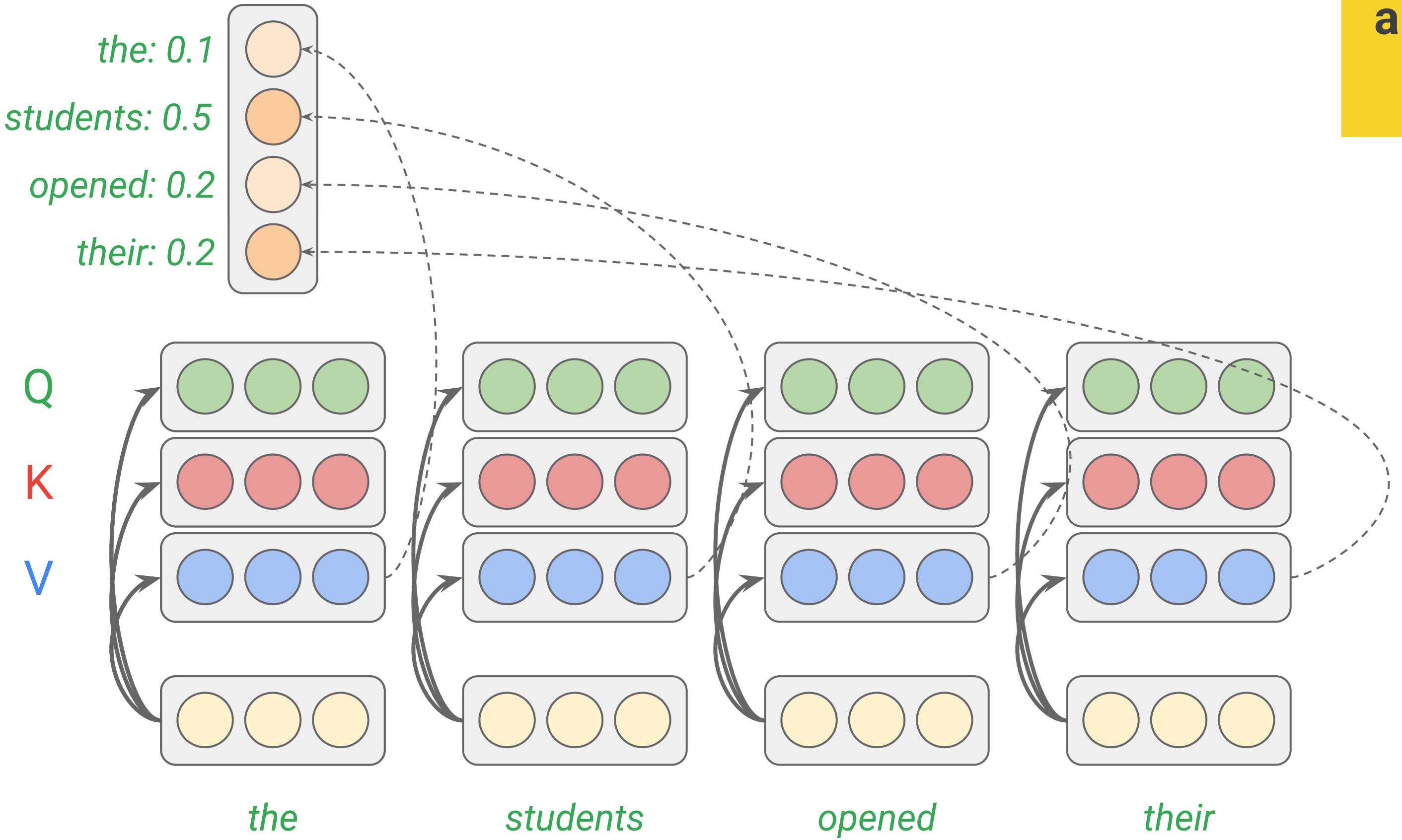
Self-attention (cont'd)

***all* computations are parallelized**



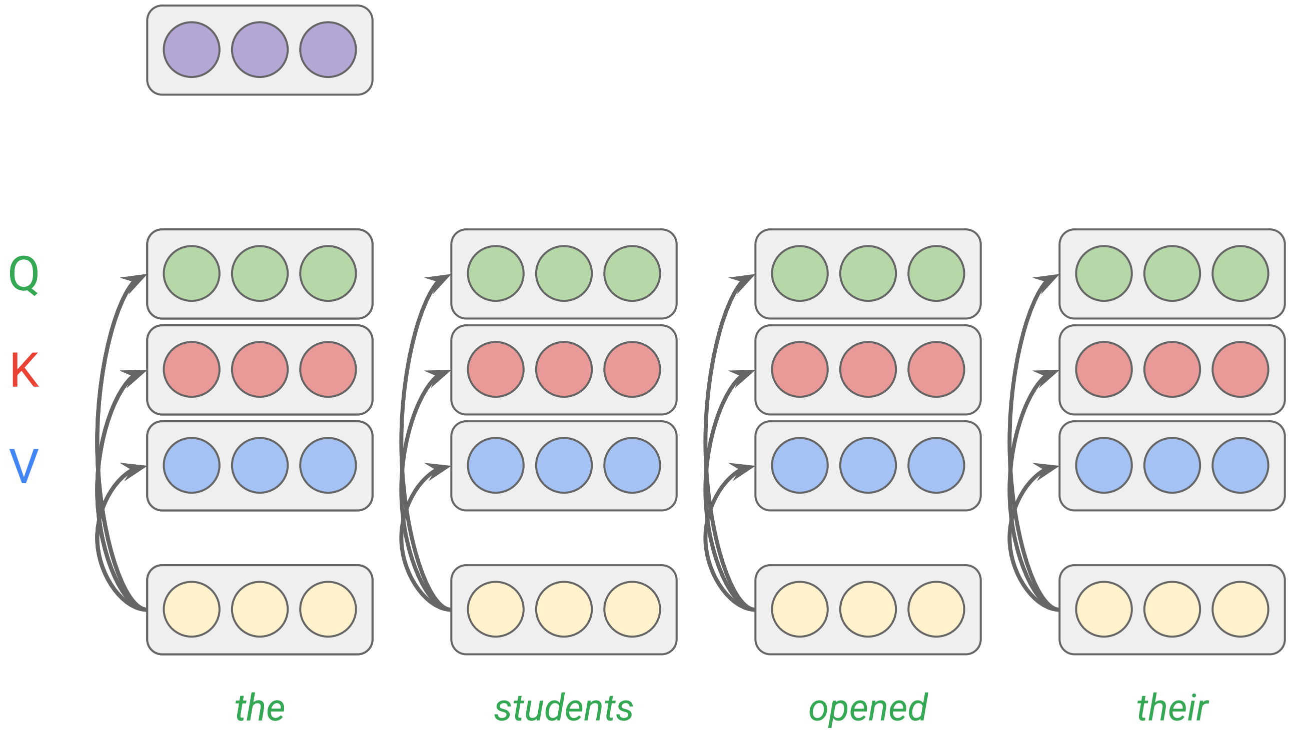
Self-attention (cont'd)

***all* computations are parallelized**



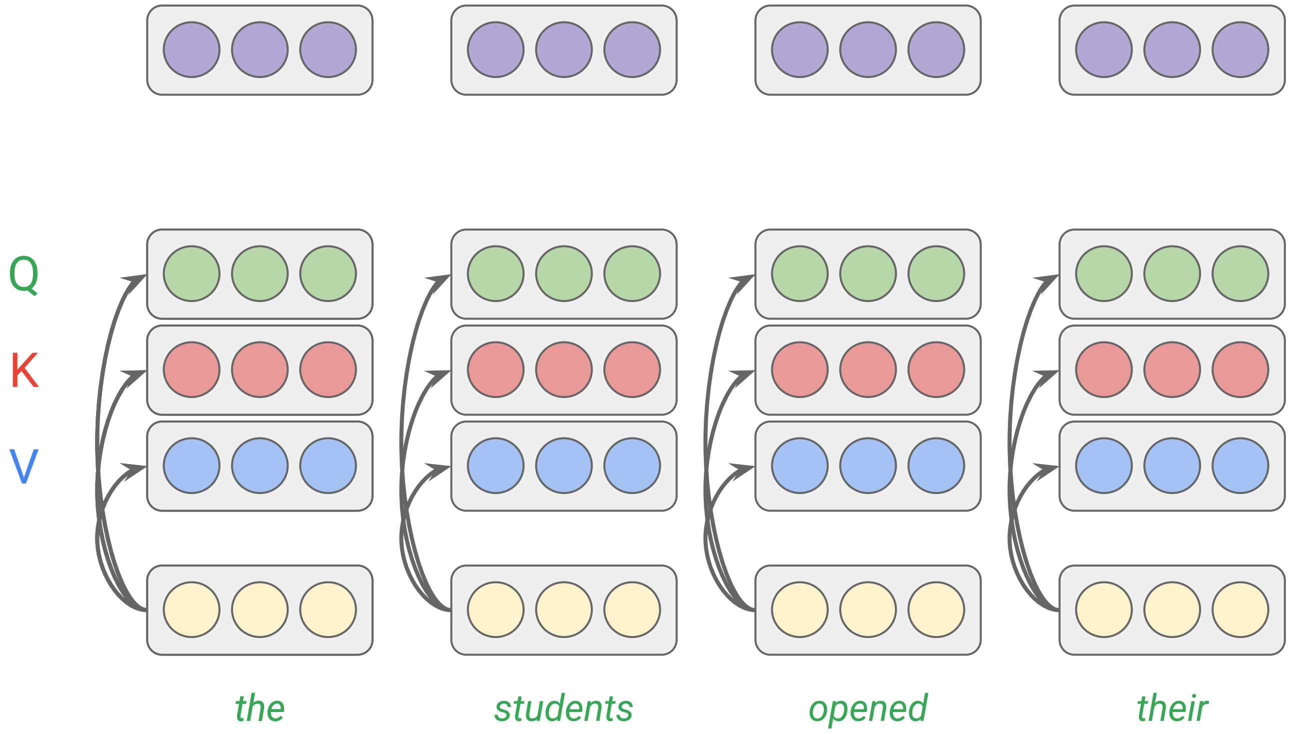
Self-attention (cont'd)

all **computations are parallelized**



Self-attention (cont'd)

all computations are parallelized during training and sequential during inference



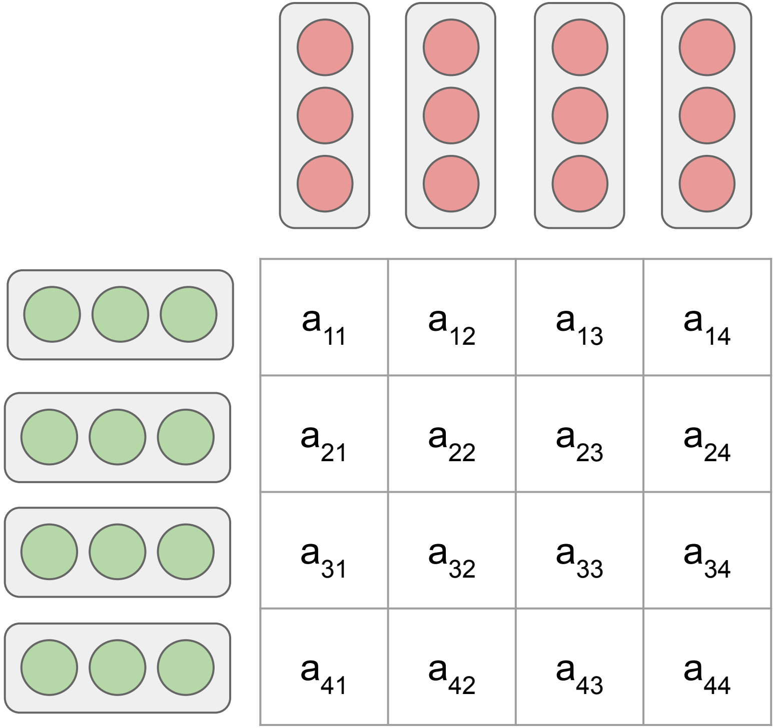
All computations are parallelized

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

d_k: scaling factor

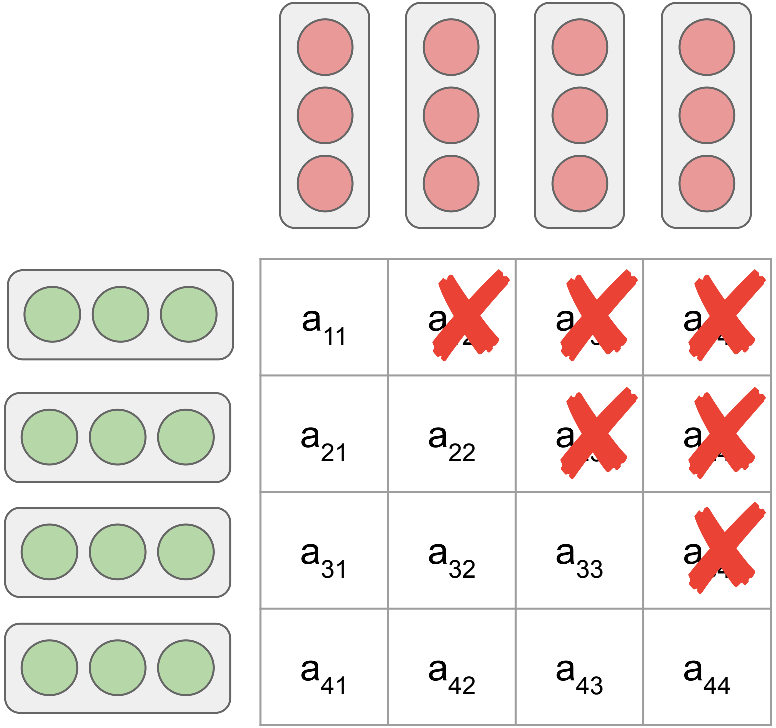
large products push the softmax function into regions where it has extremely small gradients

Quadratic complexity



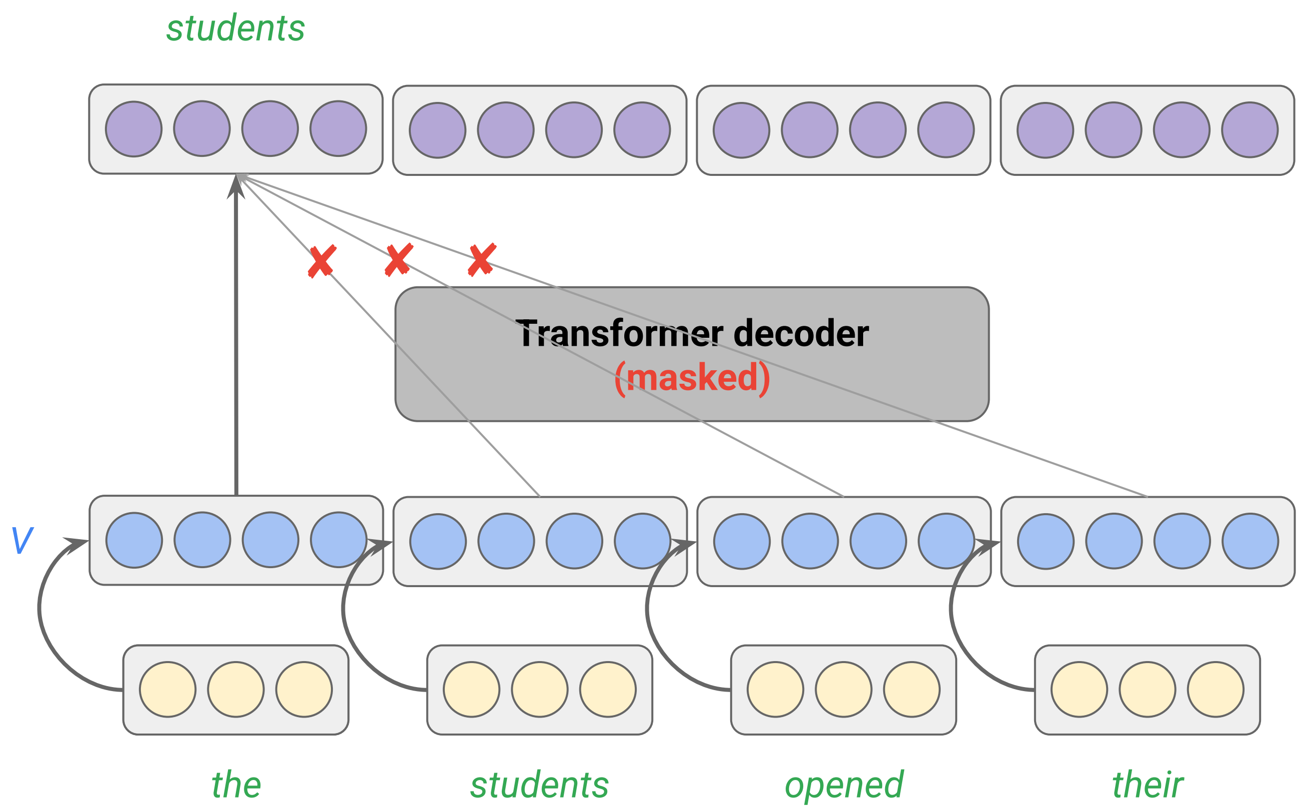
The time complexity of self-attention is quadratic in the input length $O(n^2)$

Self-attention in the decoder

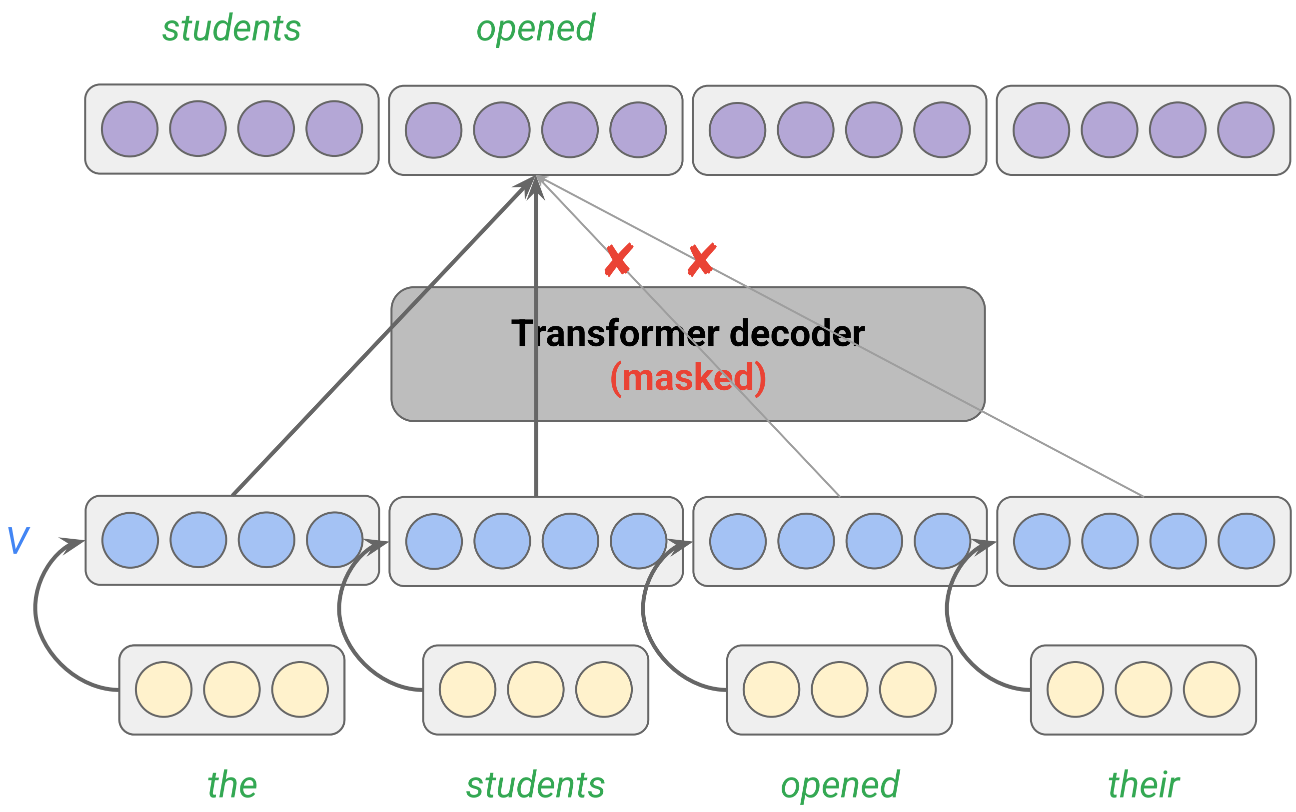


masking out all values in the input of the softmax which correspond to illegal connections

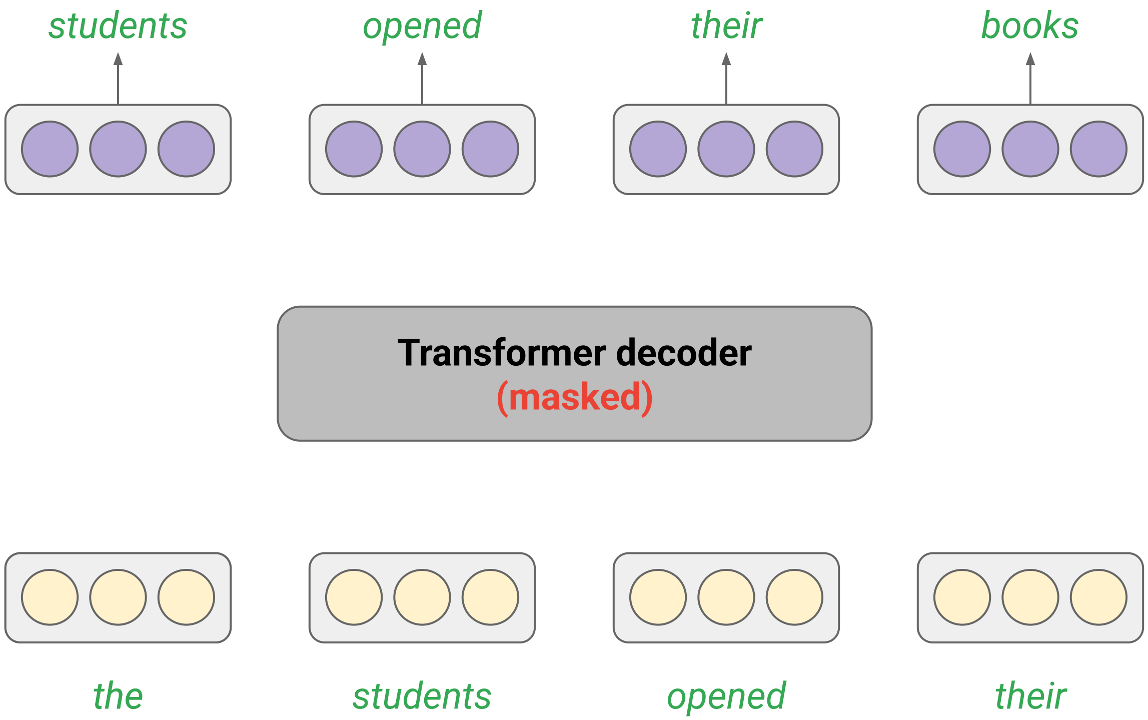
Transformer decoder



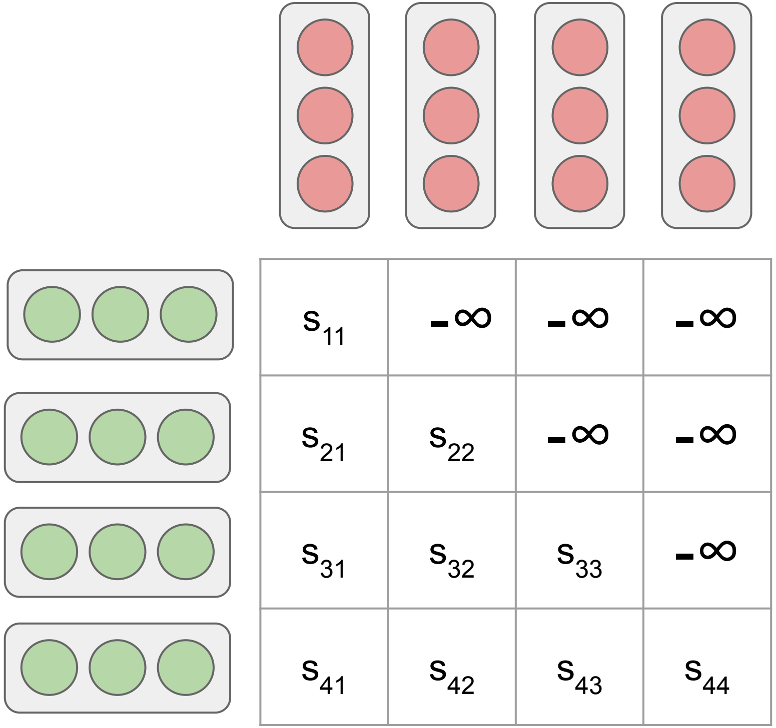
Transformer decoder (cont'd)



Transformer decoder (cont'd)



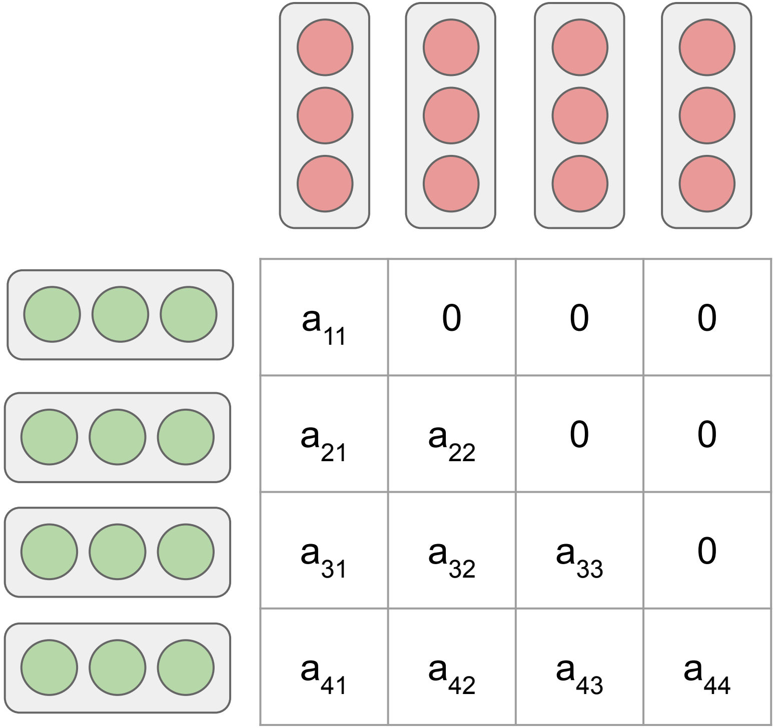
Self-attention in the decoder



s_{11}	$-\infty$	$-\infty$	$-\infty$
s_{21}	s_{22}	$-\infty$	$-\infty$
s_{31}	s_{32}	s_{33}	$-\infty$
s_{41}	s_{42}	s_{43}	s_{44}

masking out (setting to $-\infty$) all values in the input of the softmax which correspond to illegal connections

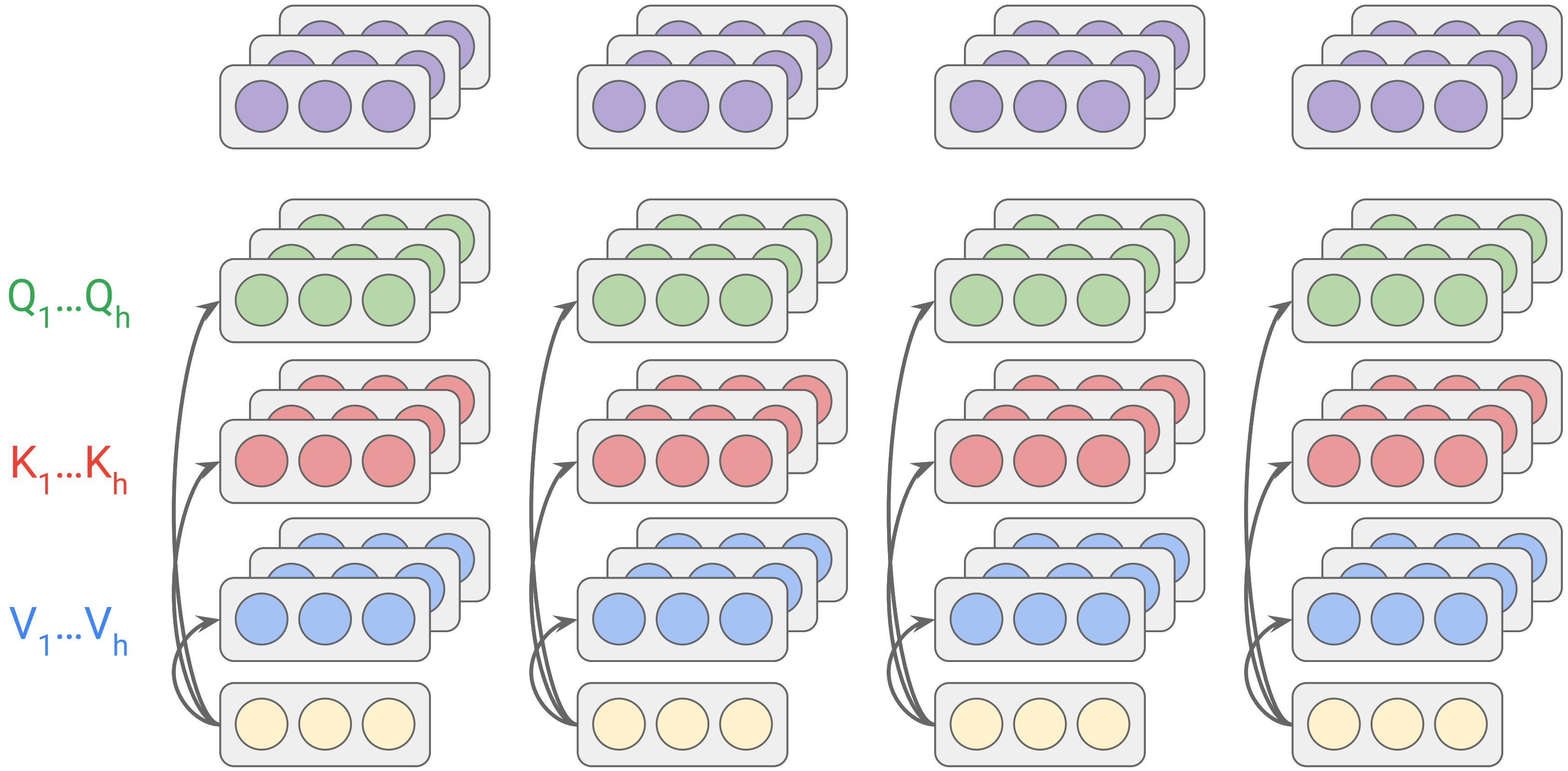
Self-attention in the decoder (cont'd)



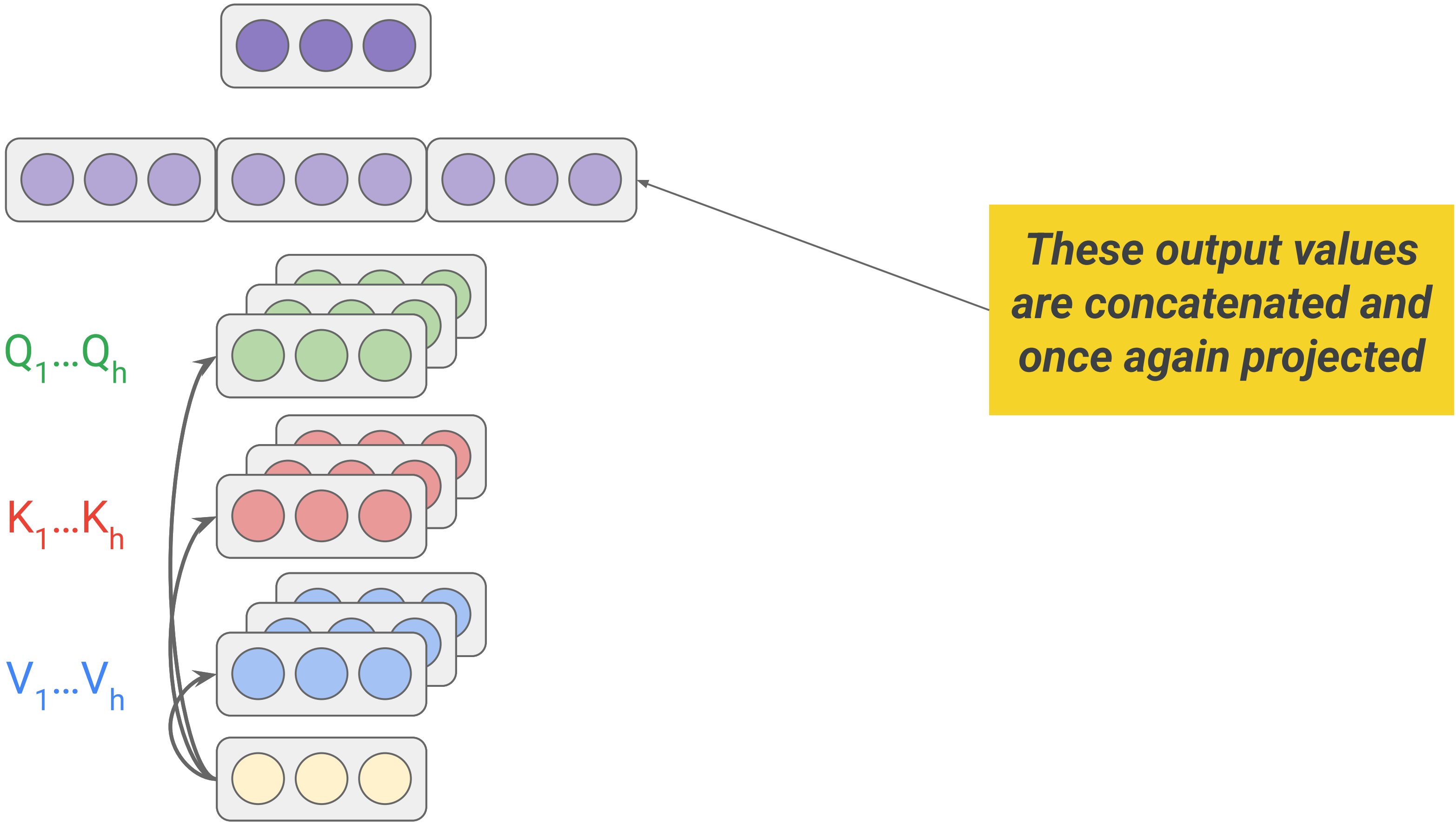
a_{11}	0	0	0
a_{21}	a_{22}	0	0
a_{31}	a_{32}	a_{33}	0
a_{41}	a_{42}	a_{43}	a_{44}

masking out all values in the input of the softmax which correspond to illegal connections

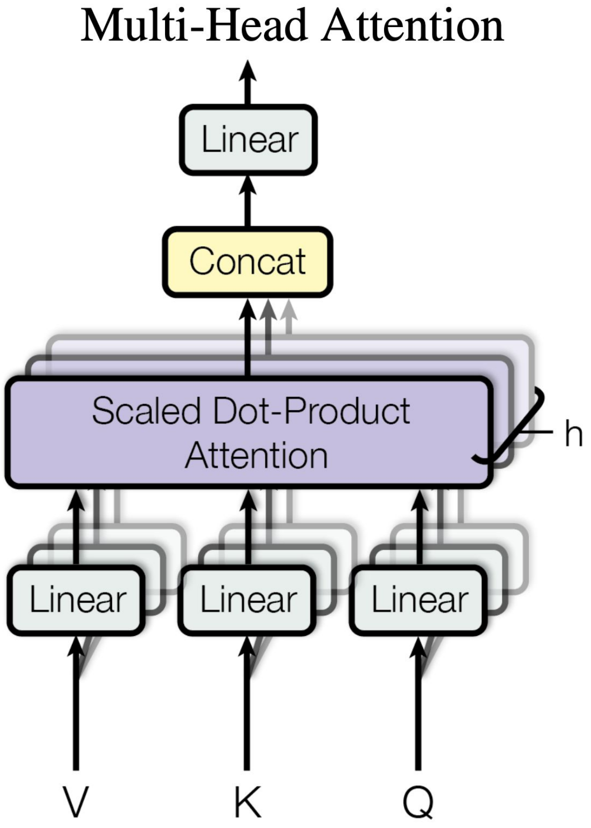
Multi-head attention



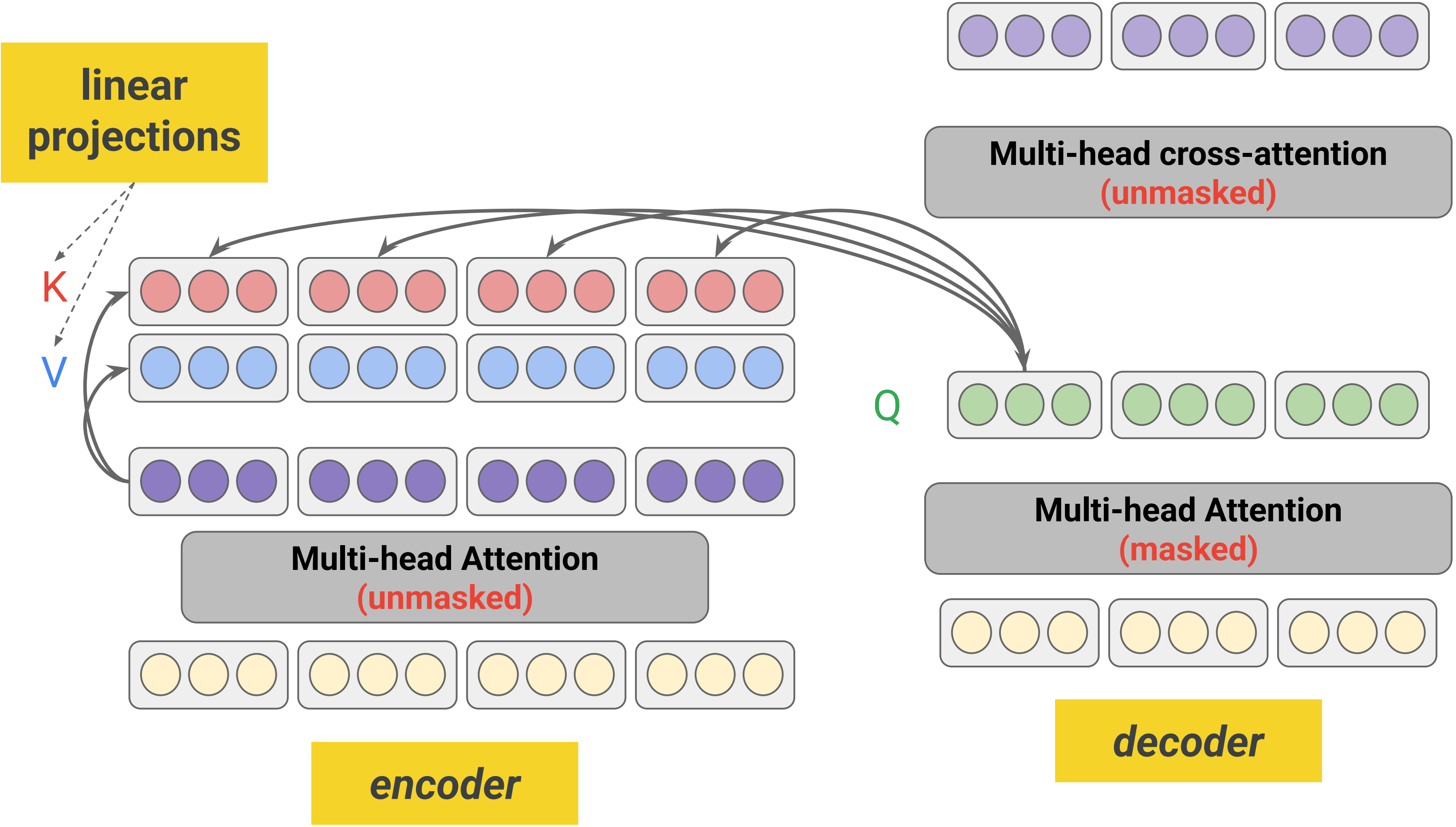
Multi-head attention (cont'd)



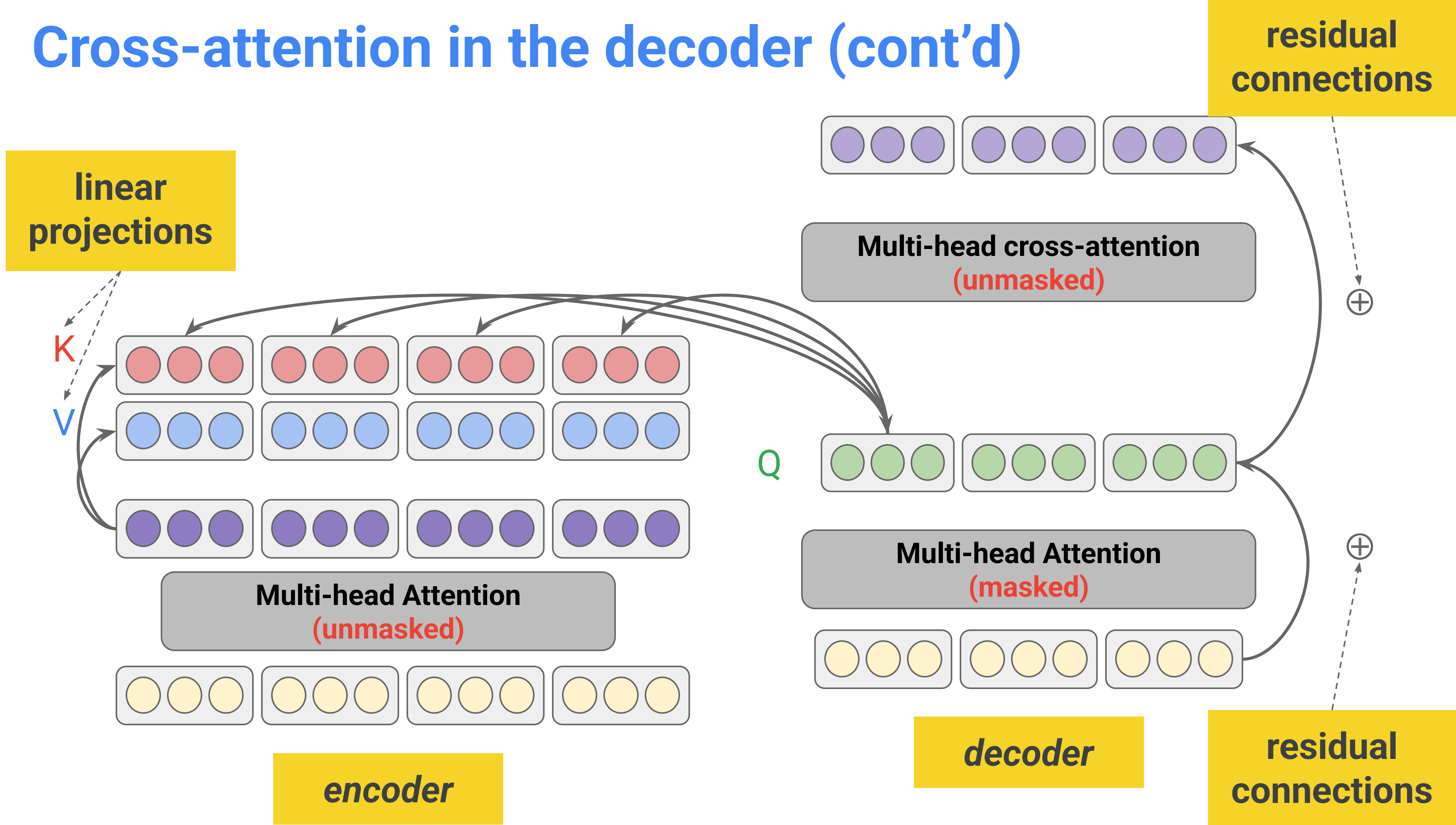
Multi-head attention (cont'd)



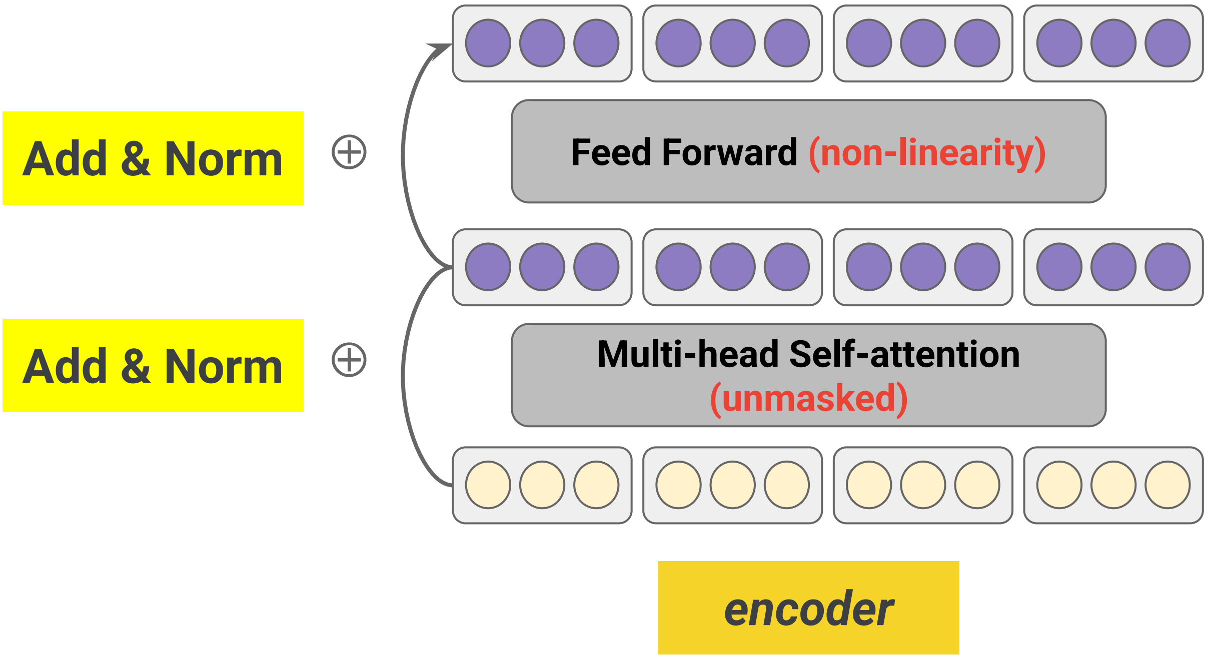
Cross-attention in the decoder



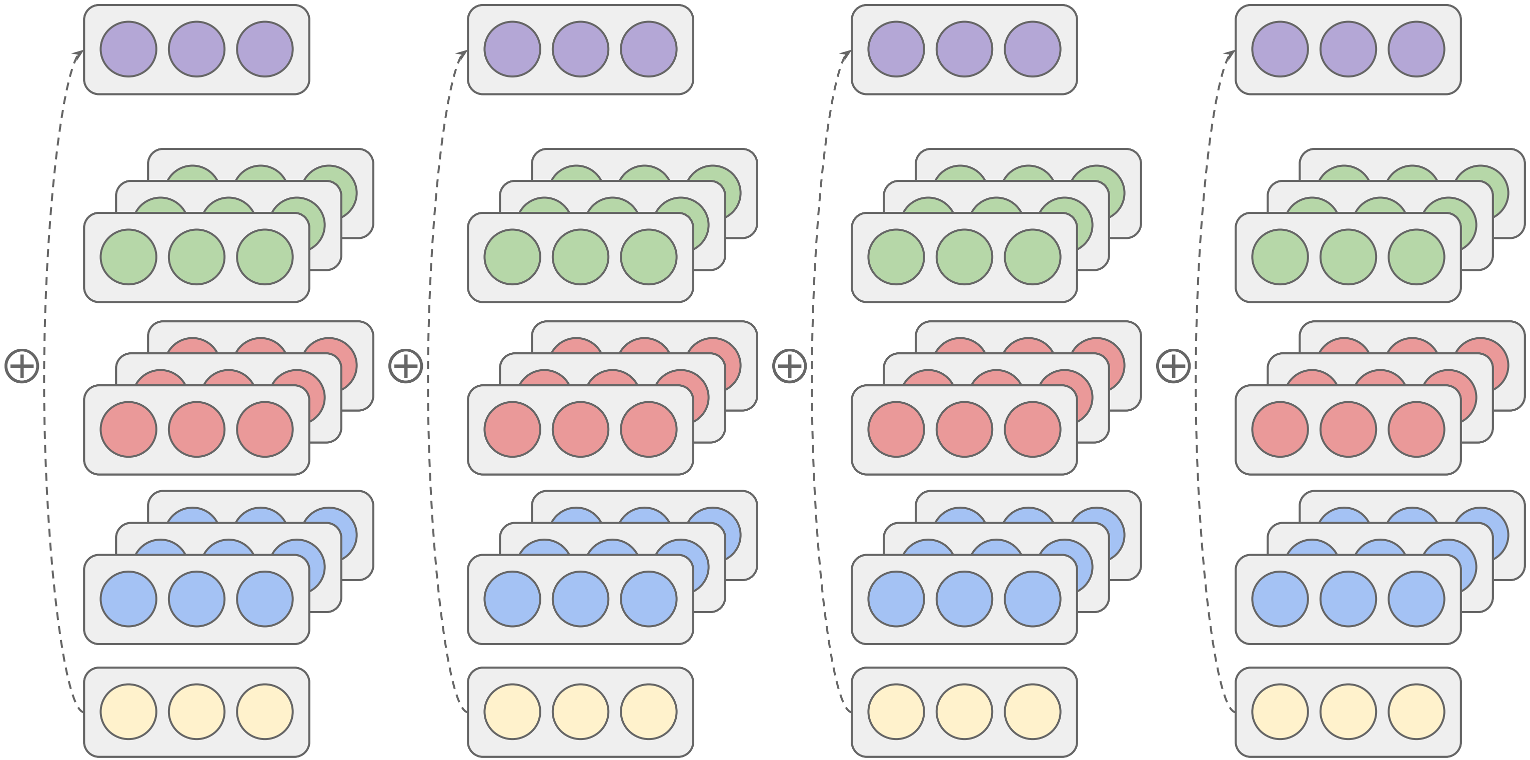
Cross-attention in the decoder (cont’d)



Encoder (one layer)



Residual connection



Residual connection

$$\text{output} = \text{sublayer}(x) + x$$

Layer normalization

$$\mathrm{Norm}(z) = \frac{z - \mu}{\sigma} \cdot \gamma + \beta$$

where μ and σ are the mean and standard deviation of the activations, and γ, β are learnable parameters.

Each activation vector is normalized so that its components have mean 0 and variance 1. This prevents activations from becoming too large or too small as they propagate through the network.

Residual connection and layer normalization

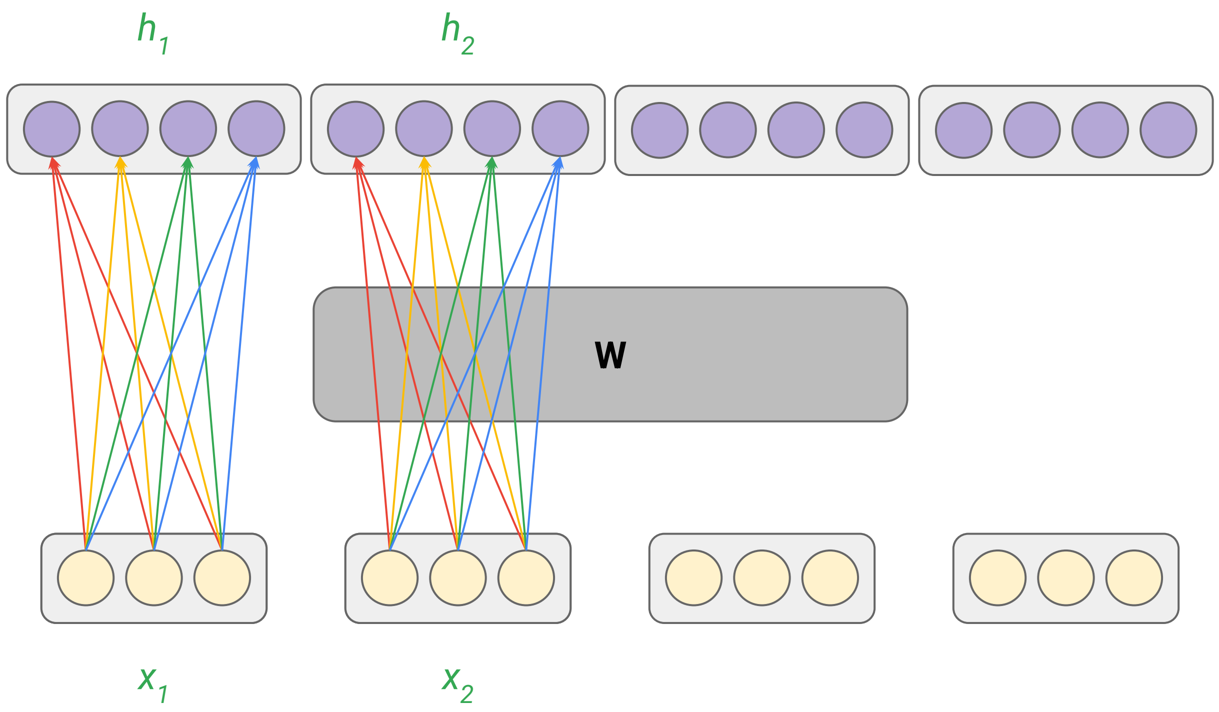
$$\mathrm{LayerNorm}(x + \mathrm{Sublayer}(x))$$

Position-wise Feed-Forward Networks

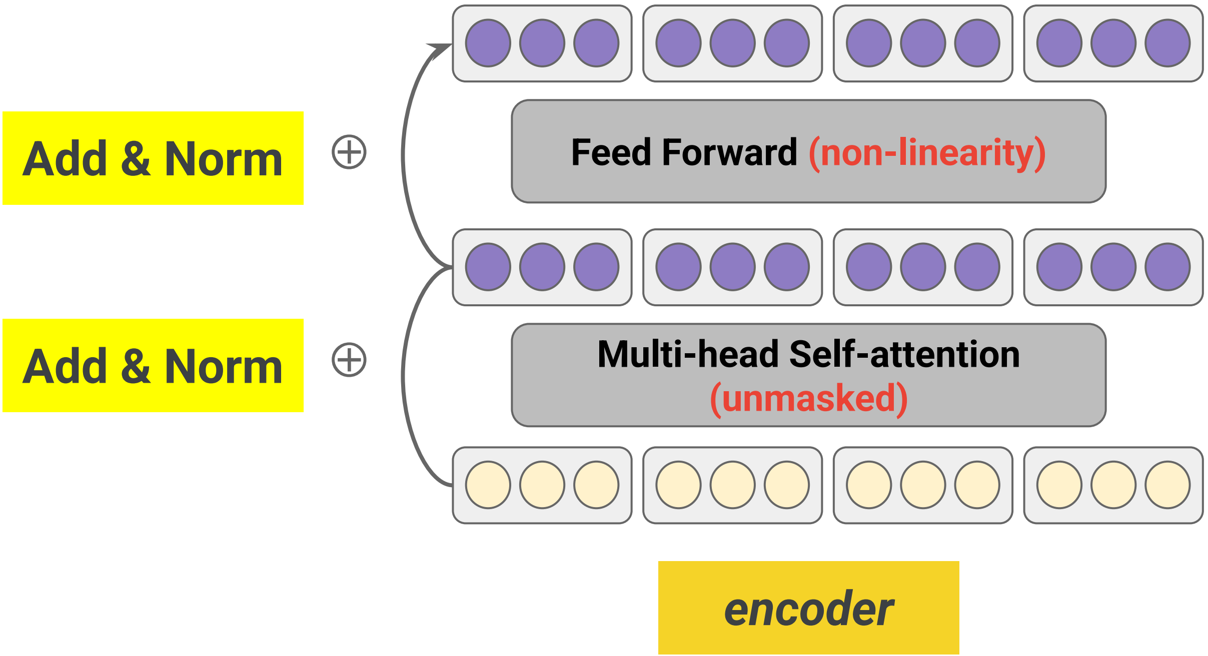
$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

ReLU (Rectified Linear Unit)

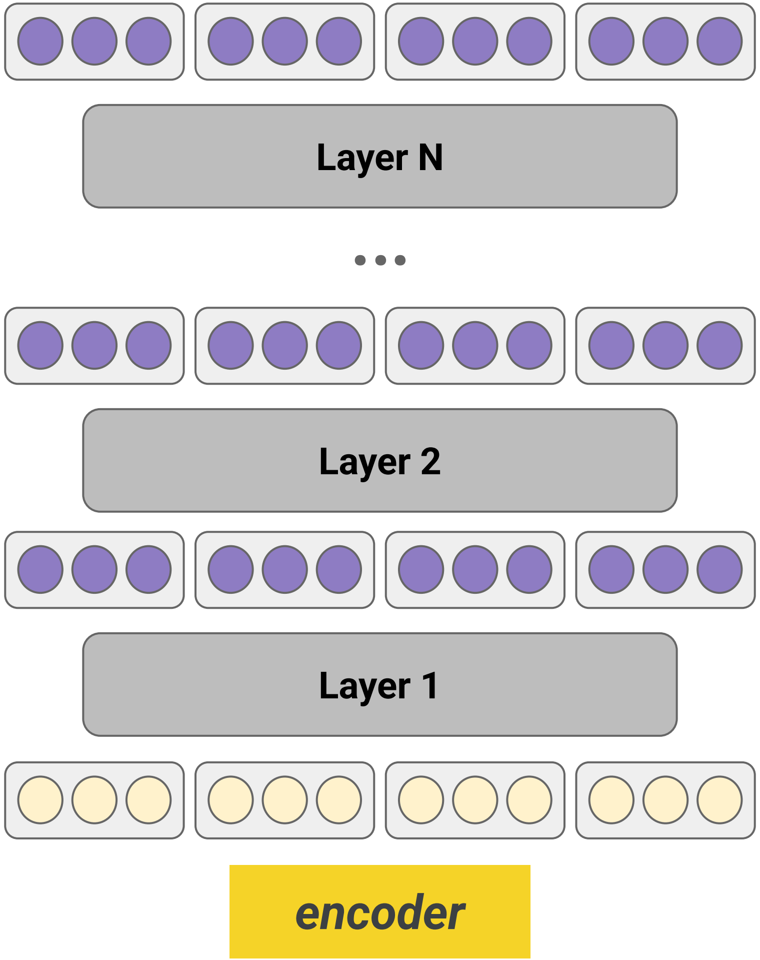
Position-wise Feed-Forward Networks (cont'd)



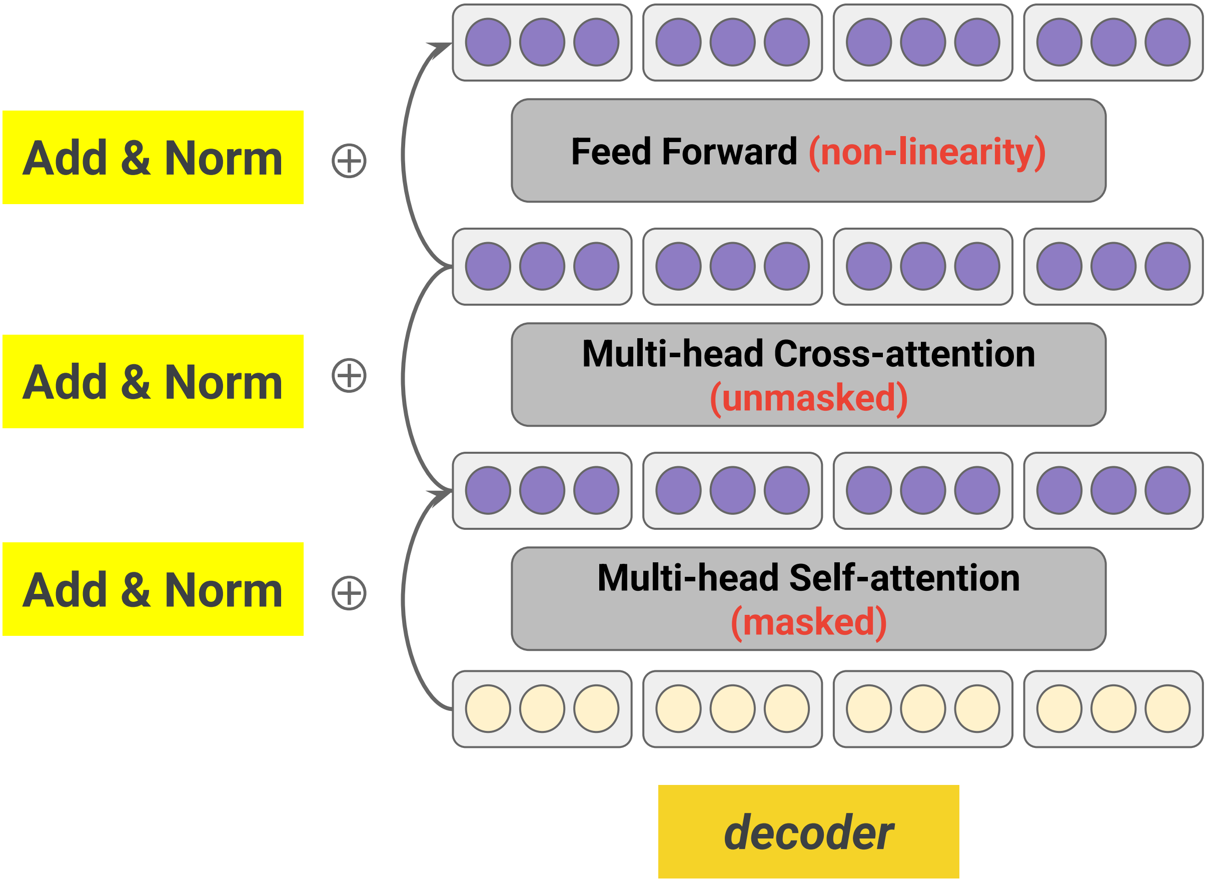
Encoder (one layer)



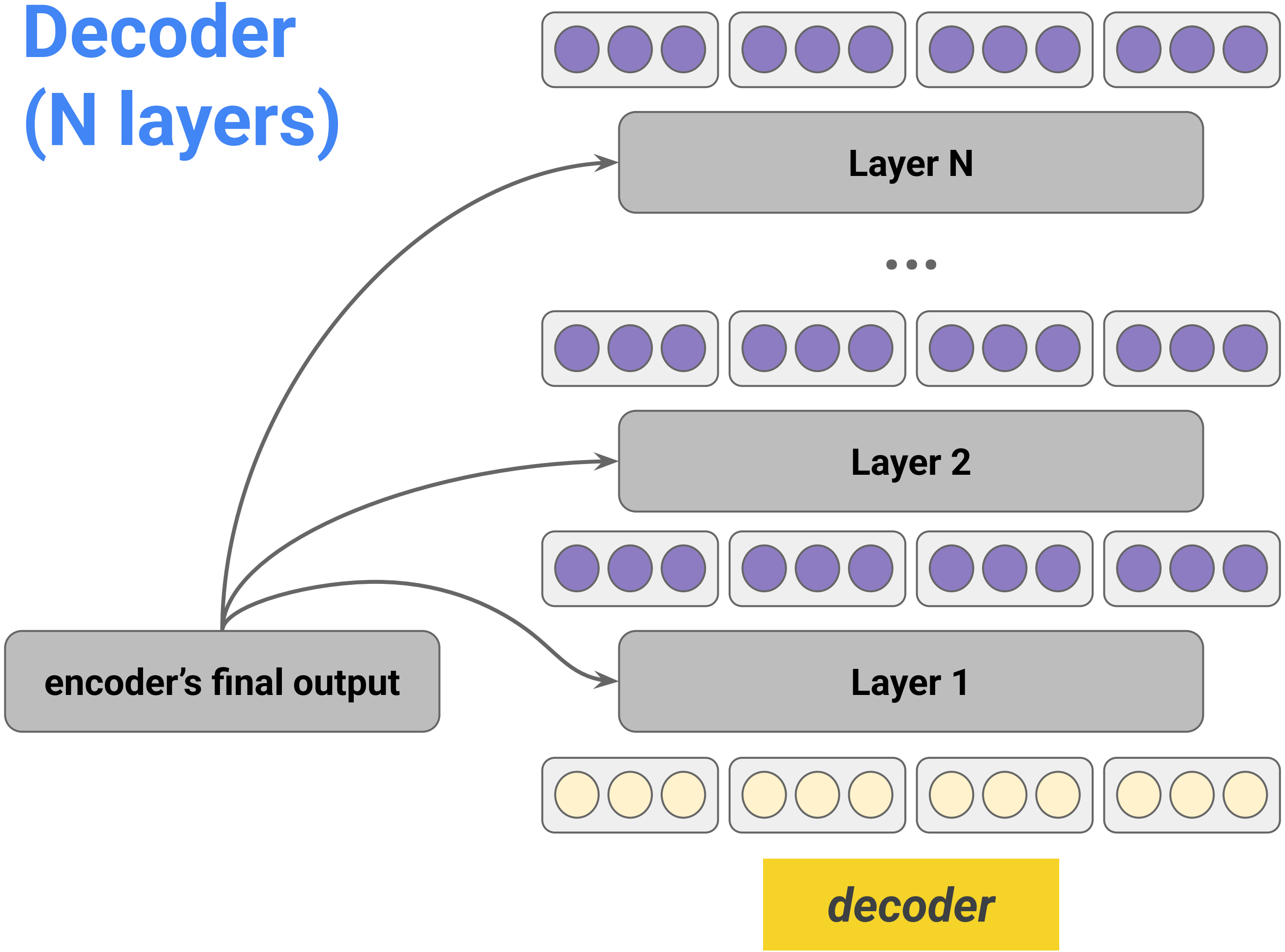
Encoder (N layers)



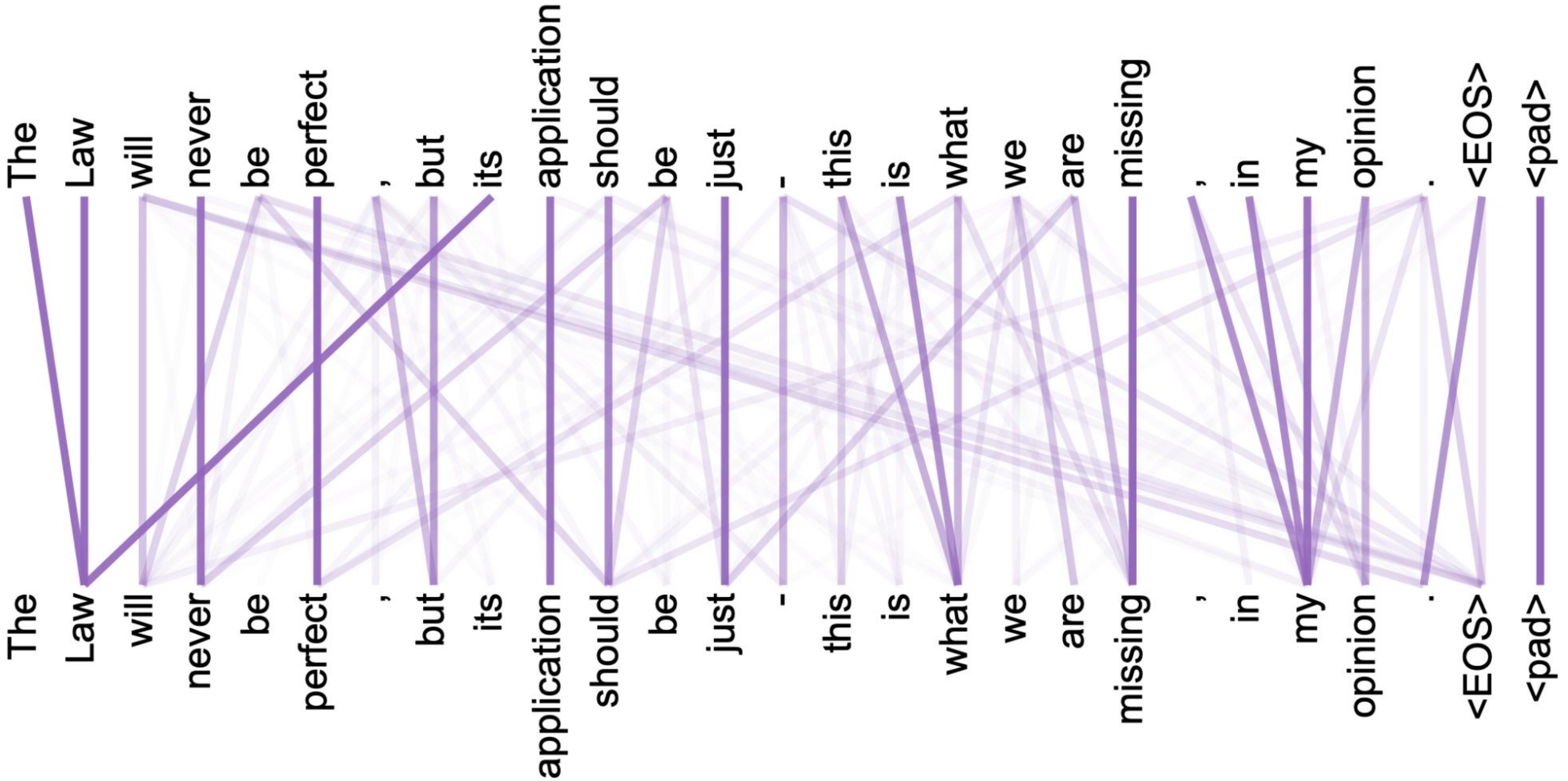
Decoder (one layer)



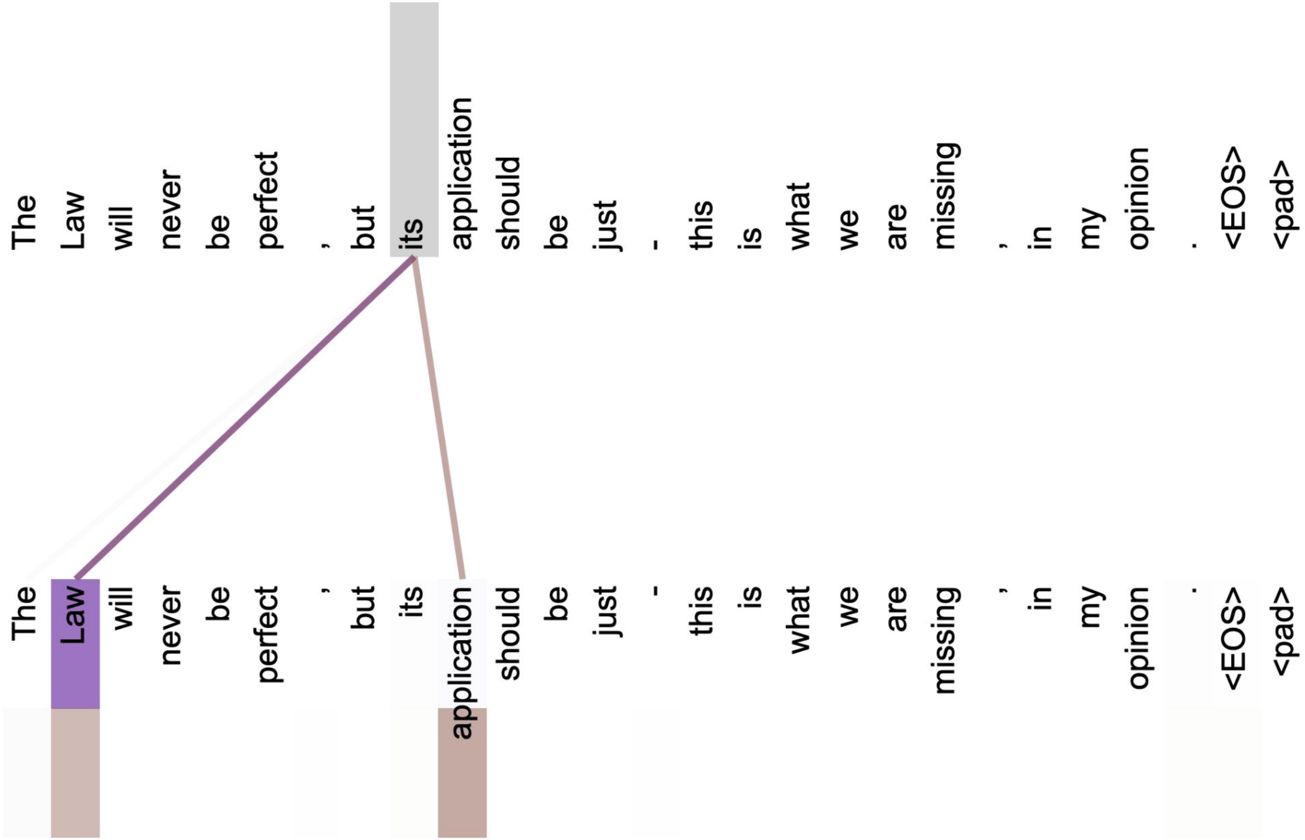
Decoder (N layers)



Attention visualizations



Attention visualizations (cont'd)

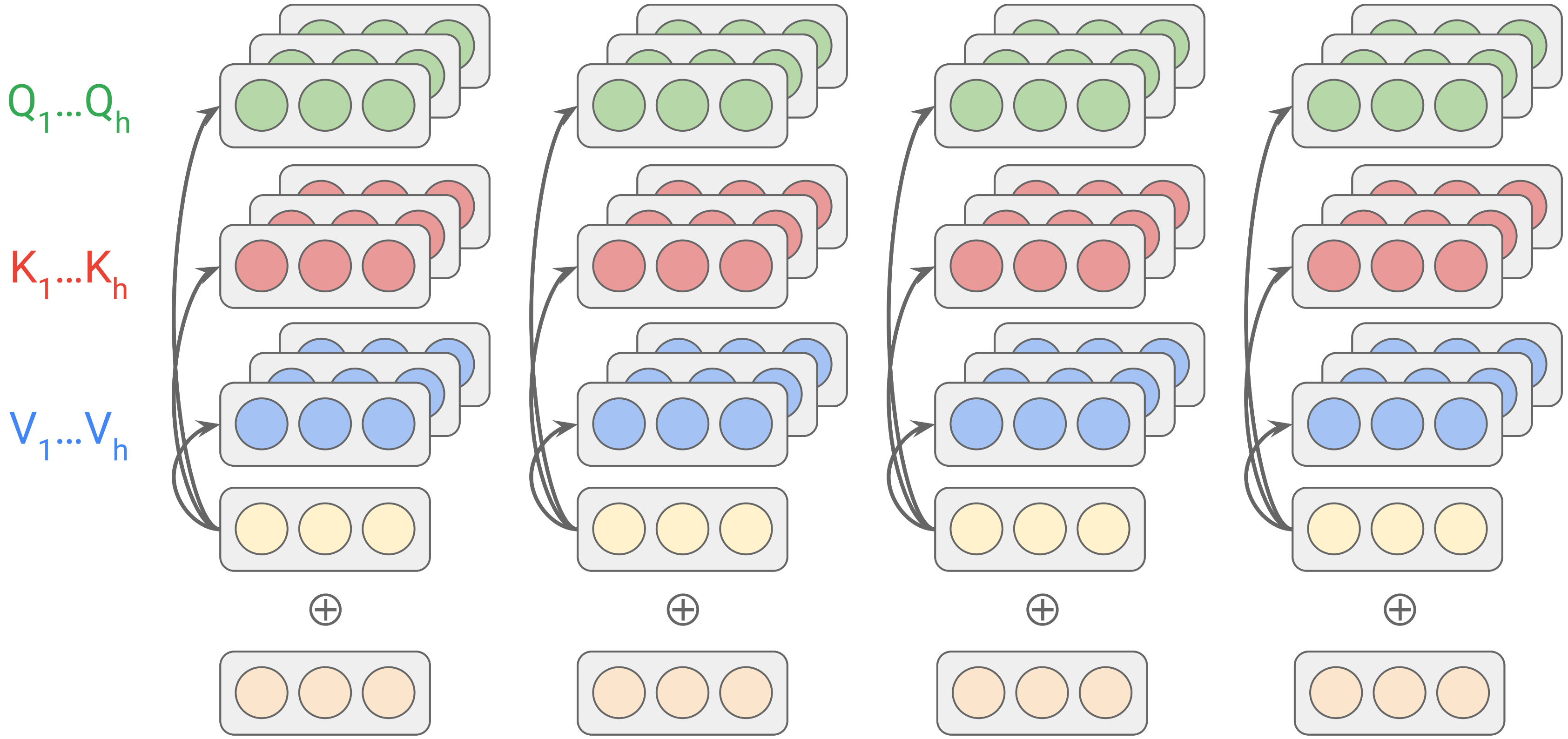


Sinusoidal positional encoding

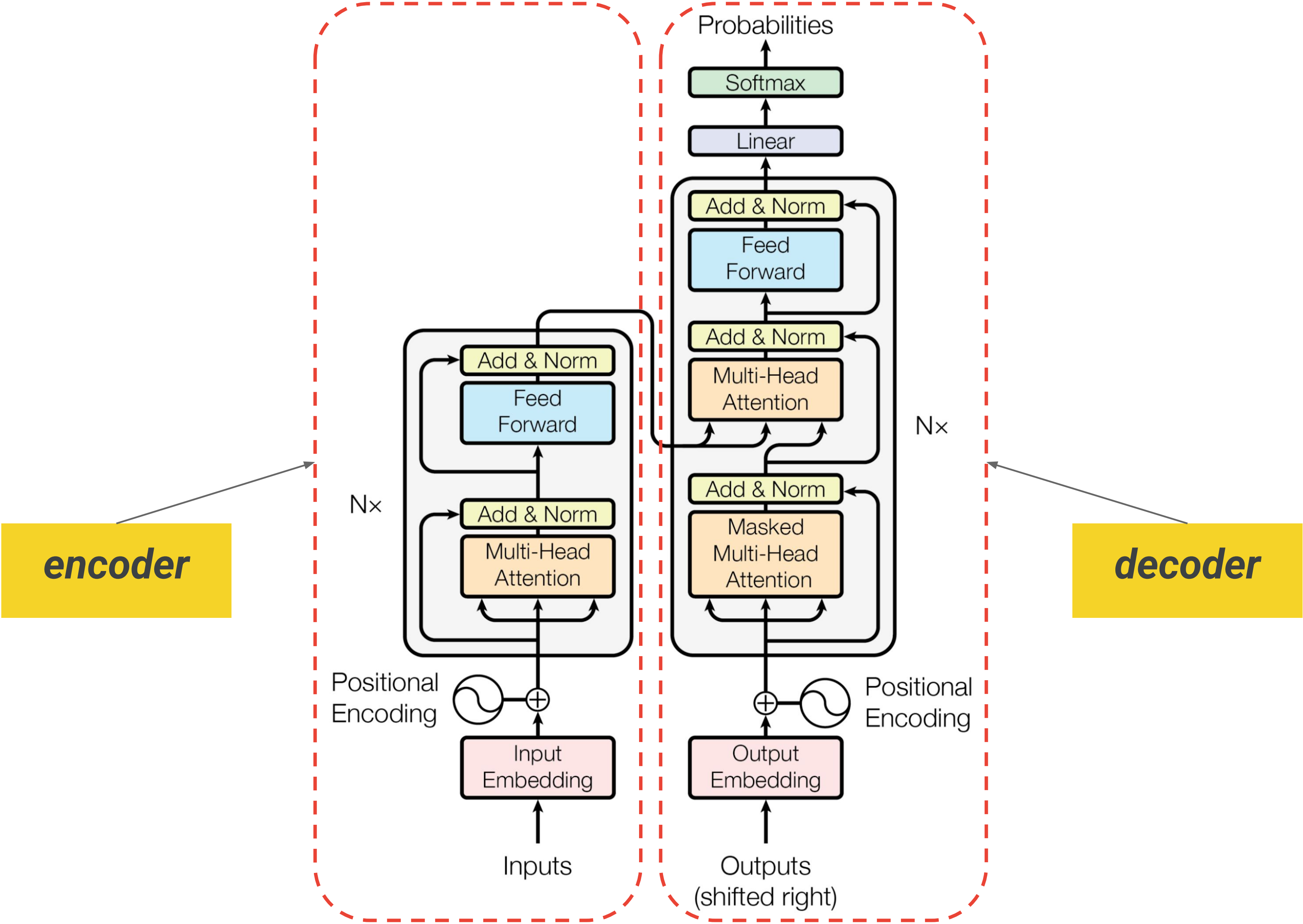
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

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

Positional Encoding (cont’d)



Transformer block (putting it together)


encoder
decoder
Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

Thank you!