Transformers (cont'd)

CS 4804: Introduction to AI

Fall 2025

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

Tu Vu

VIRGINIA TECH

Logistics

- Homework 0 (due today)
- Final project groups finalized
  - Project proposal template released today (due in 3 weeks)

  - Project research ideas
    - Check out GPT-5 System Card & Gemini 2.5 Pro Technical Report
    - New benchmarks
    - New prompting methods
    - Improving small models
    - …

Useful libraries

- Hugging Face (https://huggingface.co)
  - Transformers
  - Models / Datasets
- Unsloth (https://unsloth.ai)
  - Efficient fine-tuning
- vLLM (https://github.com/vllm-project/vllm)
  - Efficient inference
- Together.AI
  - APIs for Open-weight models

vLLM

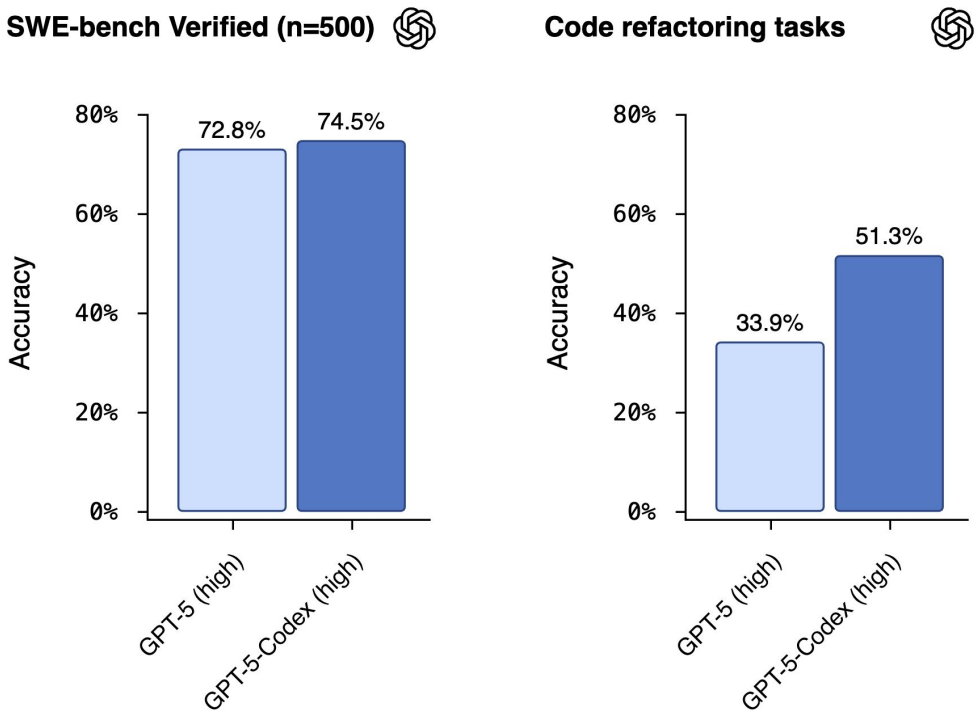
together.ai

AI News

- Google’s Nano Banana 🍌🍌🍌
  - restore old photos
  - try out different hairstyles
  - imagine yourselves as 3D model figurines

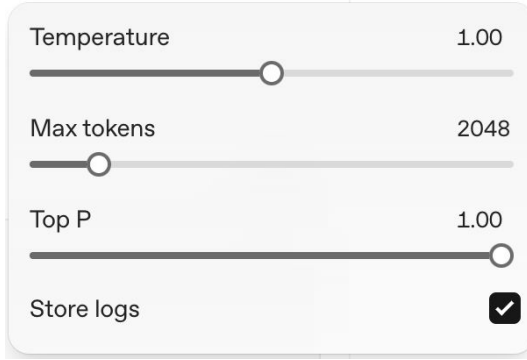
AI News (cont'd)

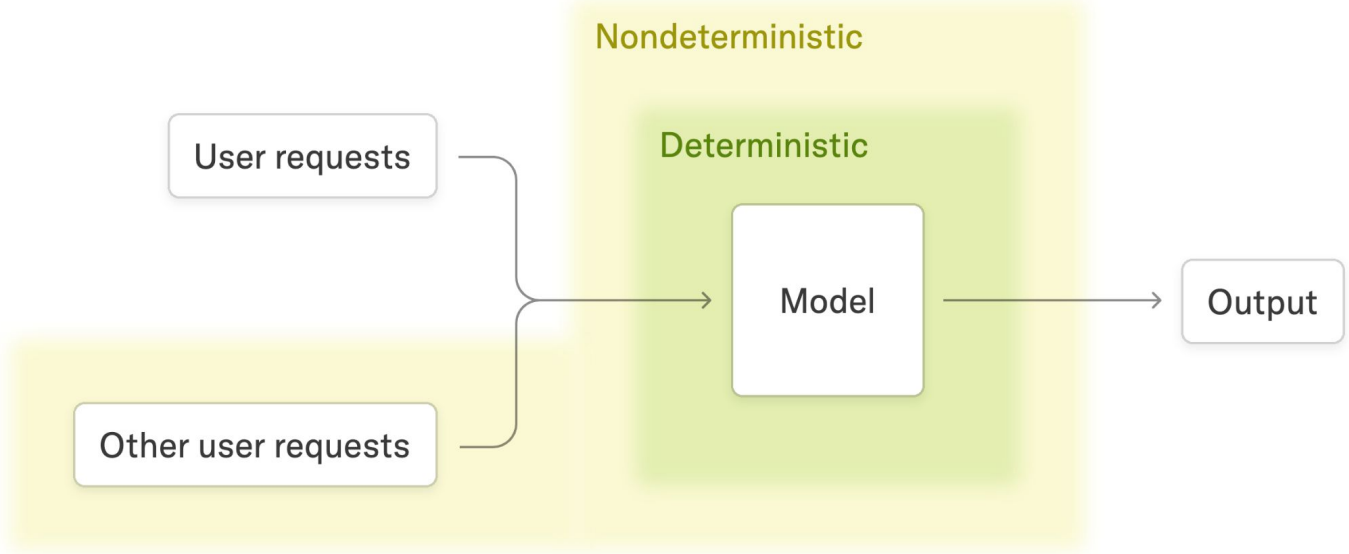
- OpenAI's GPT-5-Codex: GPT-5 optimized for agentic coding



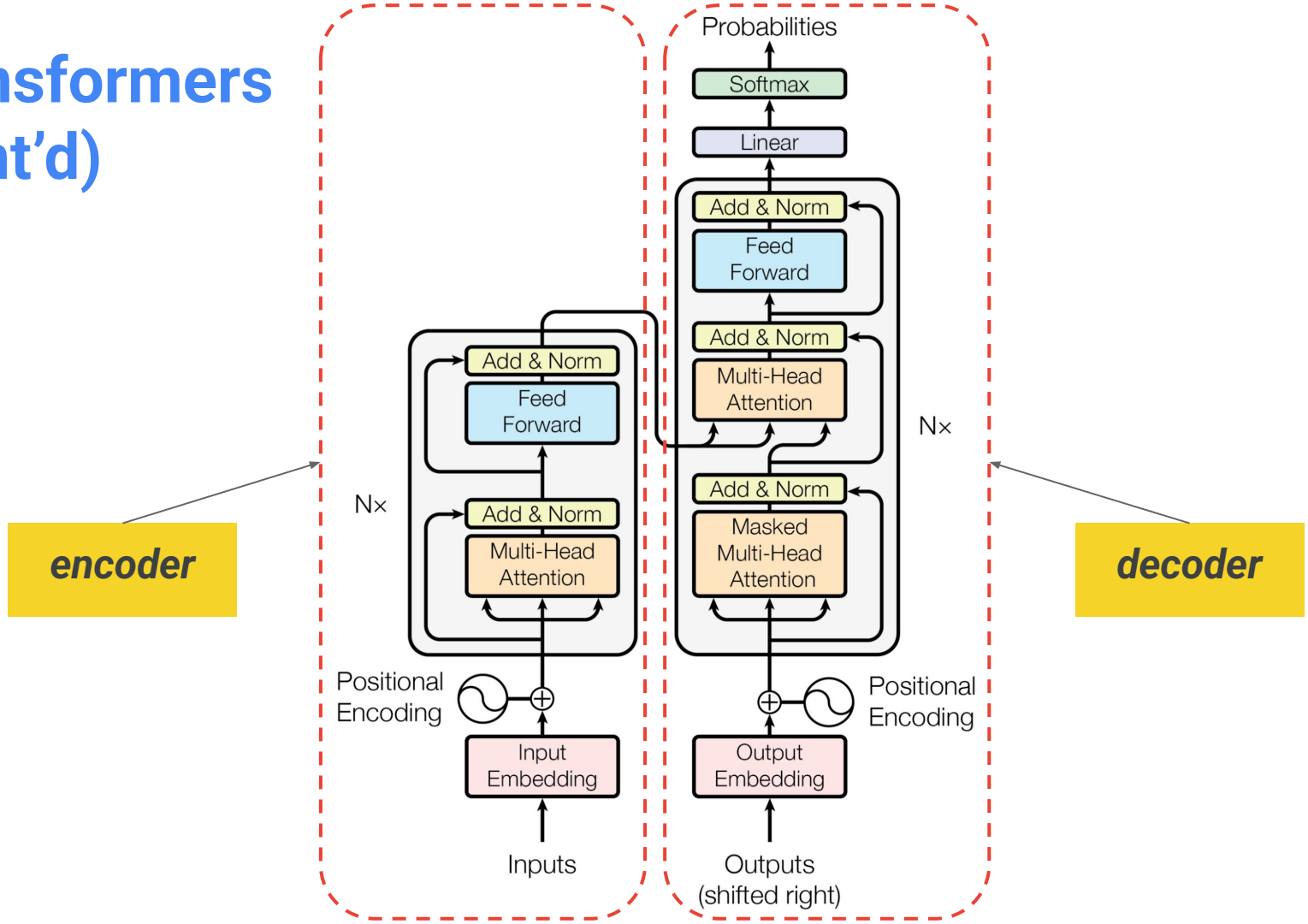
AI News (cont'd): Thinking Machines

- Defeating Nondeterminism in LLM Inference

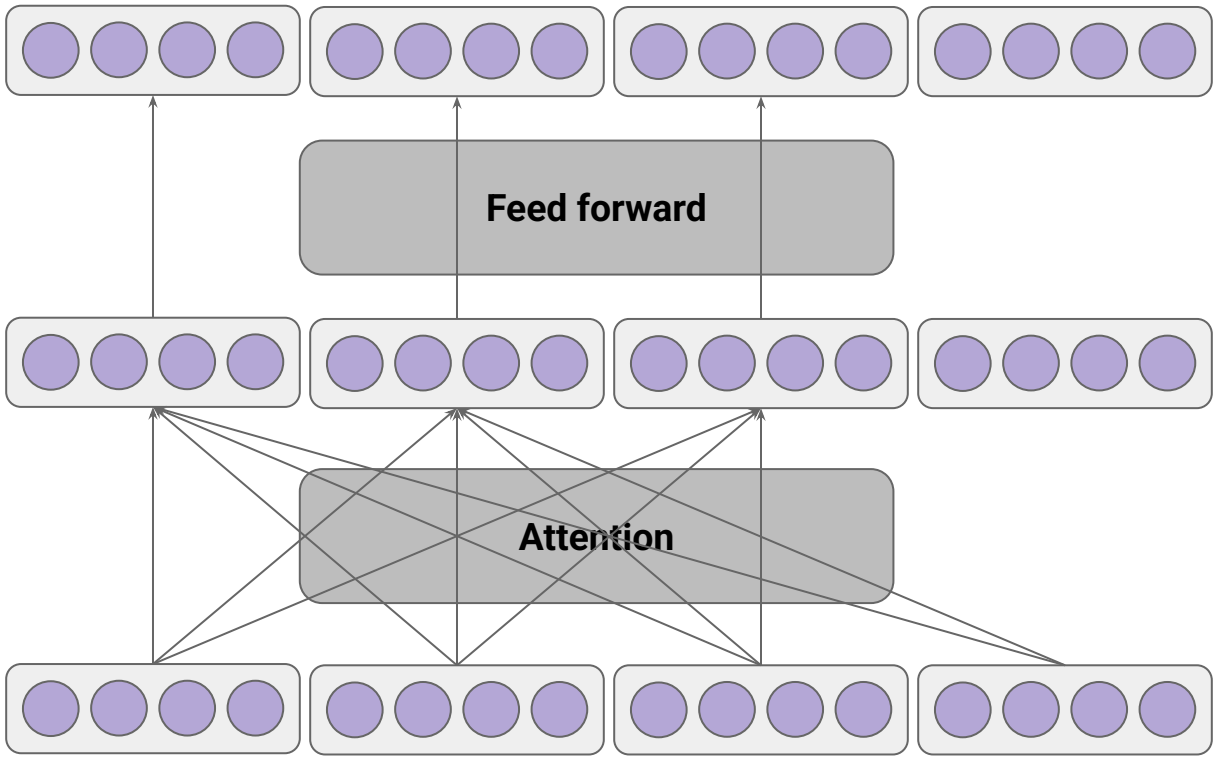




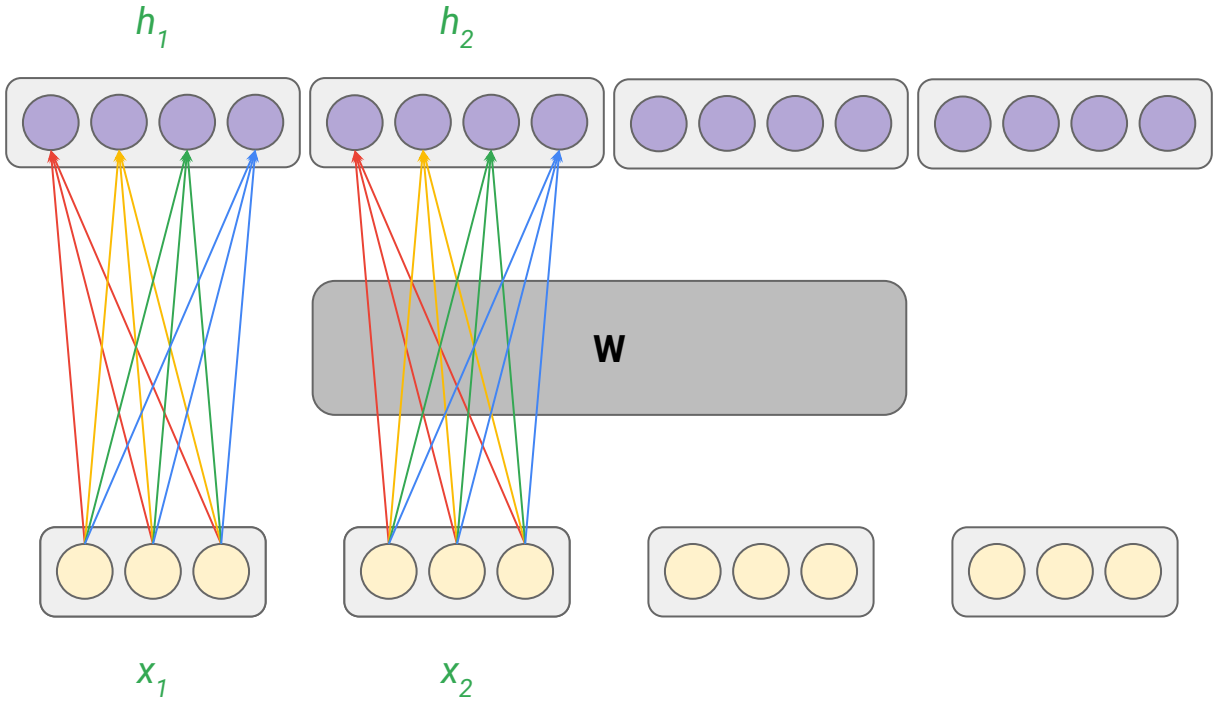
Transformers (cont'd)



Encoder



Position-wise Feed-Forward Networks



We multiply the weight matrix W (size 4×3) with the embeddings matrix X (size 3×2):

$$H = WX$$

Performing the multiplication:

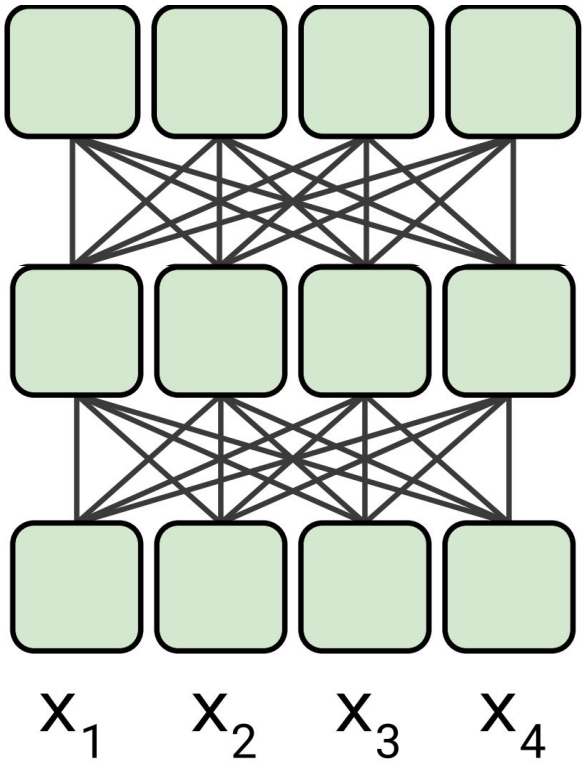
$$H = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{bmatrix} \begin{bmatrix} x_{11} & x_{21} \\ x_{12} & x_{22} \\ x_{13} & x_{23} \end{bmatrix}$$

This results in:

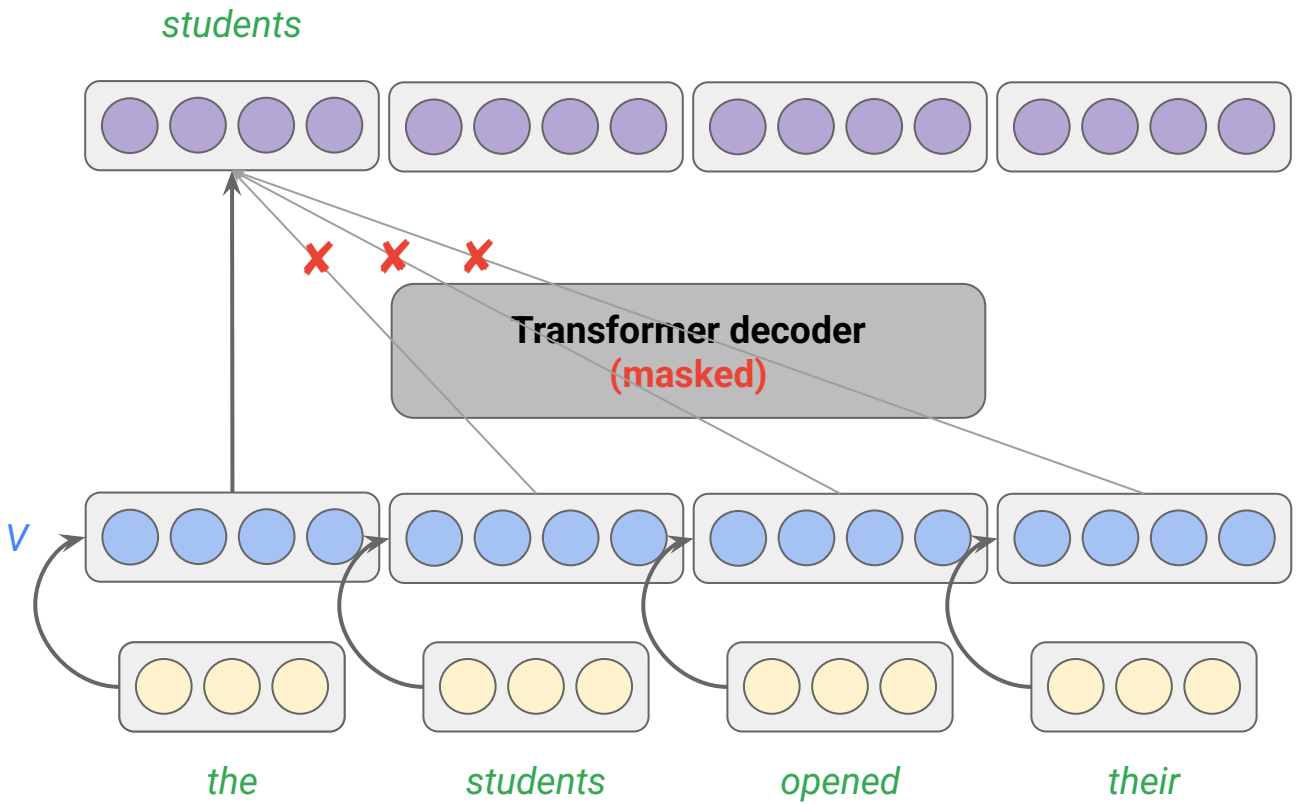
h_1 h_2

$$H = \begin{bmatrix} w_{11}x_{11} + w_{12}x_{12} + w_{13}x_{13} & w_{11}x_{21} + w_{12}x_{22} + w_{13}x_{23} \\ w_{21}x_{11} + w_{22}x_{12} + w_{23}x_{13} & w_{21}x_{21} + w_{22}x_{22} + w_{23}x_{23} \\ w_{31}x_{11} + w_{32}x_{12} + w_{33}x_{13} & w_{31}x_{21} + w_{32}x_{22} + w_{33}x_{23} \\ w_{41}x_{11} + w_{42}x_{12} + w_{43}x_{13} & w_{41}x_{21} + w_{42}x_{22} + w_{43}x_{23} \end{bmatrix}$$

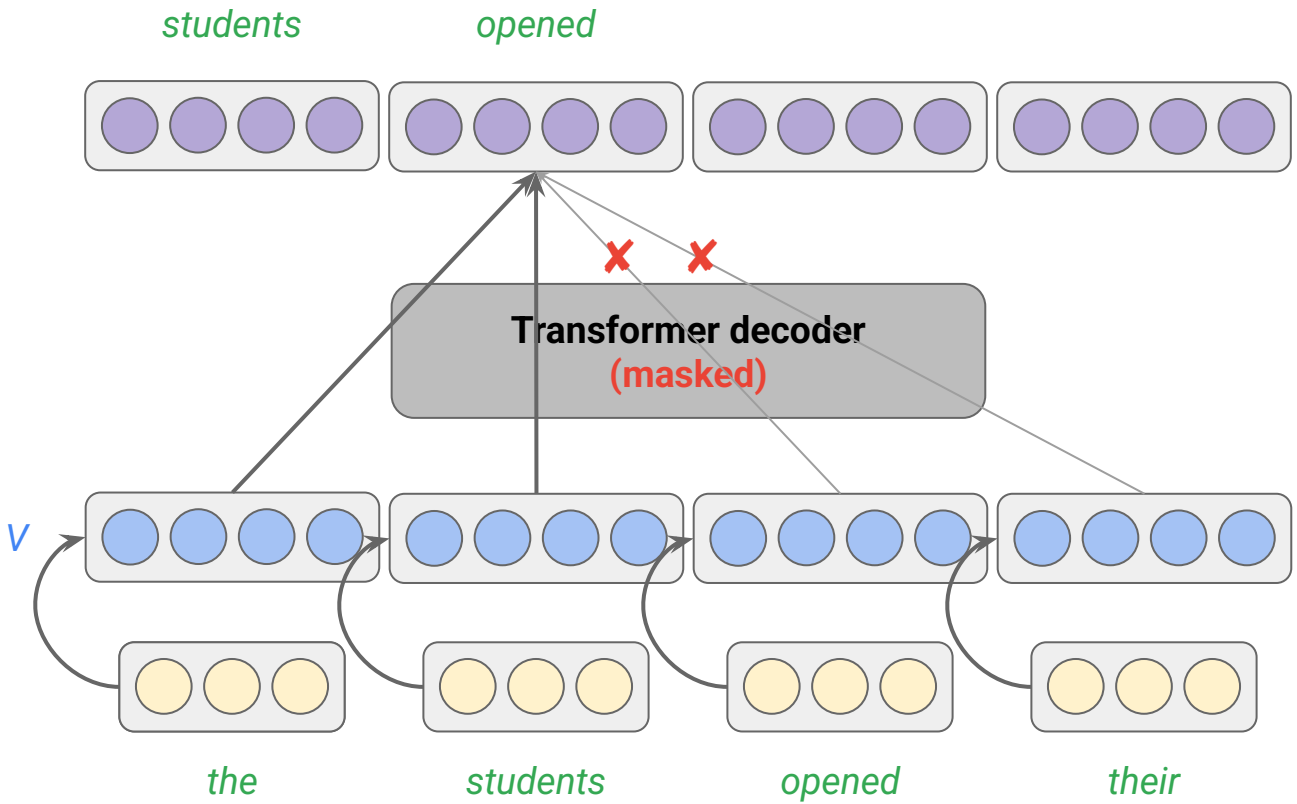
Encoder (cont'd)



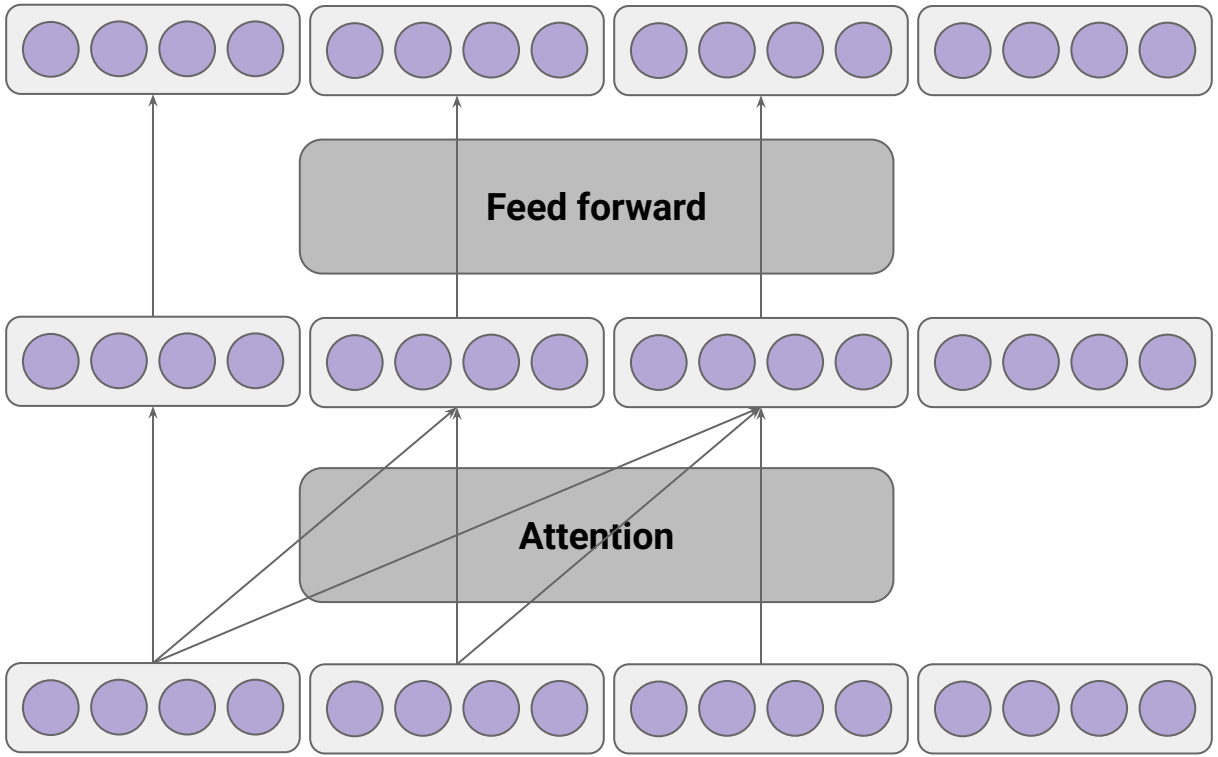
Decoder


students
Transformer decoder
(masked)
v
the
students
opened
their

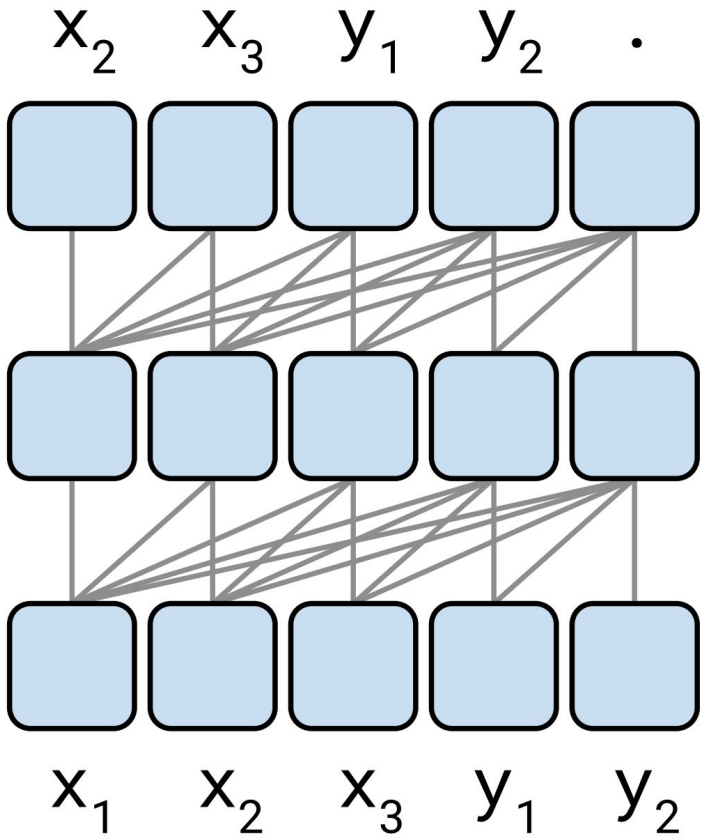
Decoder (cont'd)



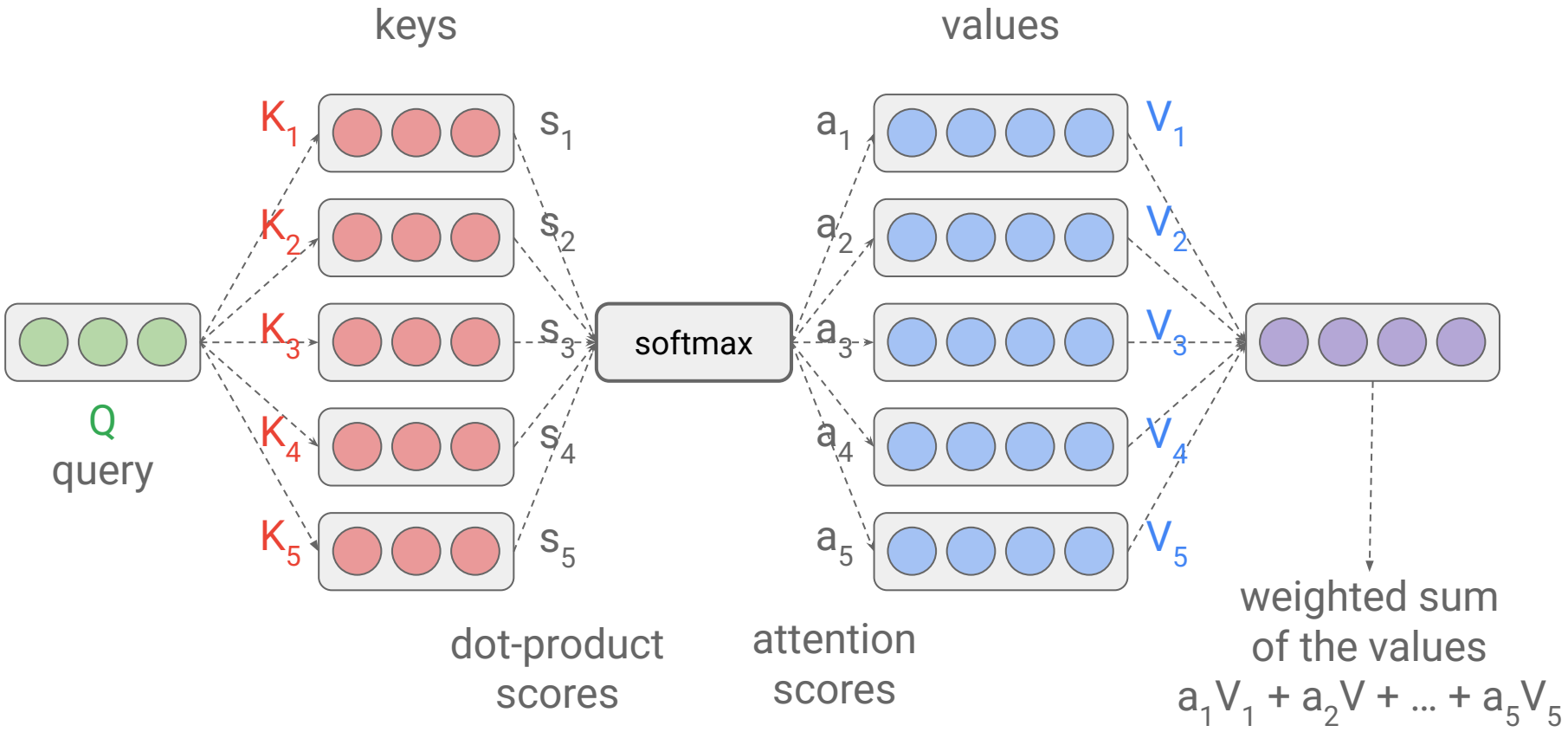
Decoder (cont'd)



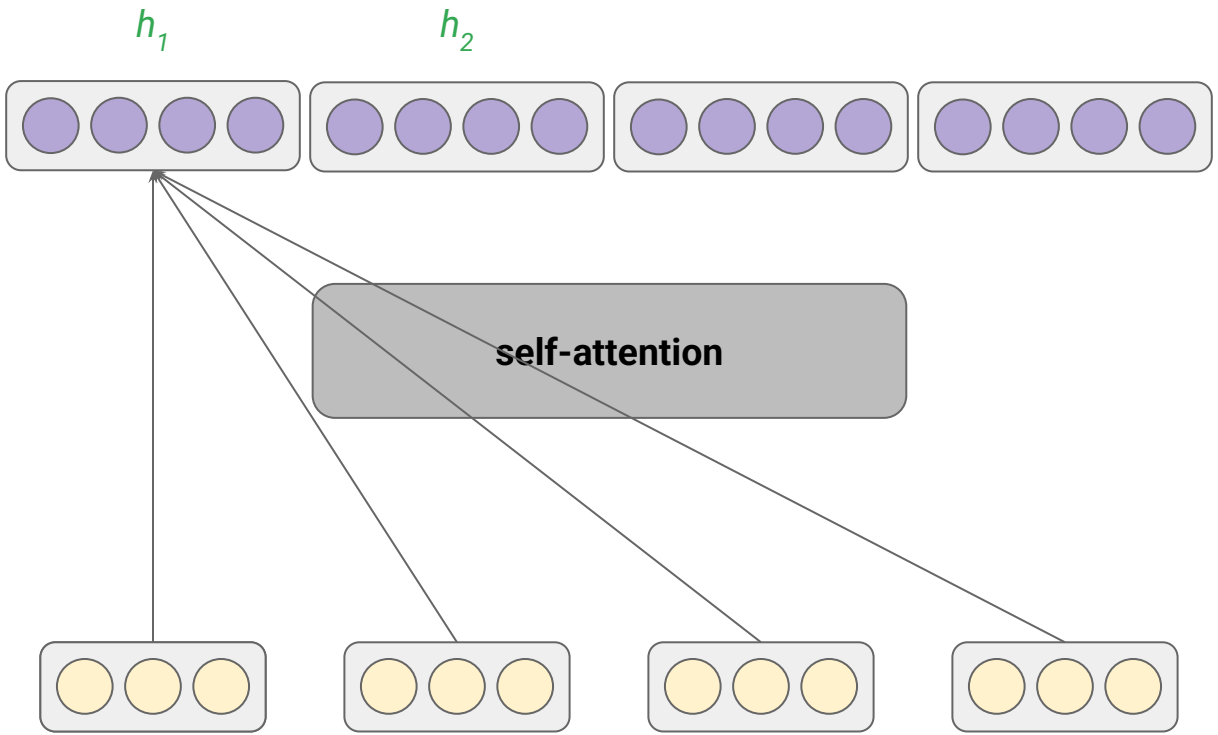
Decoder (cont'd)



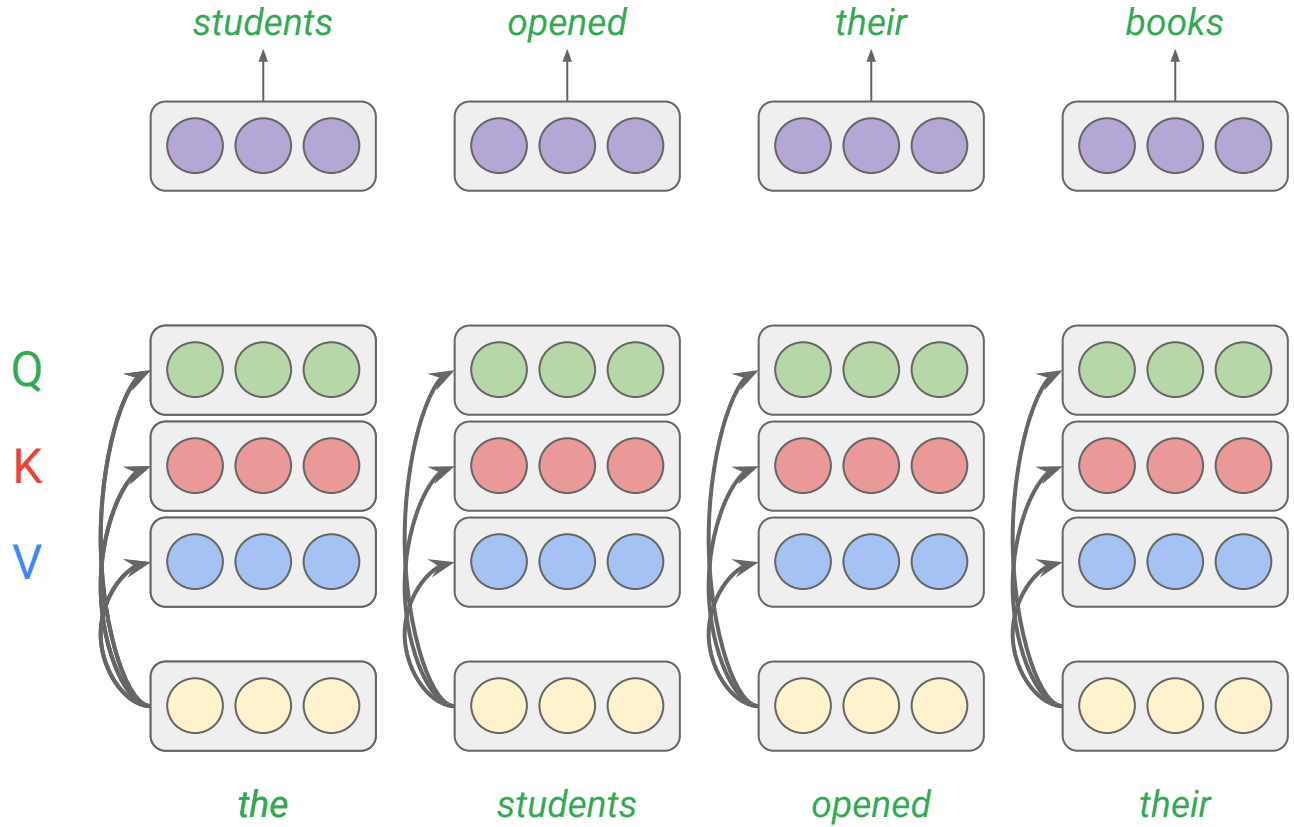
Attention mechanism



Self-attention



Self-attention (cont'd)



$$Q = X \cdot W_Q$$

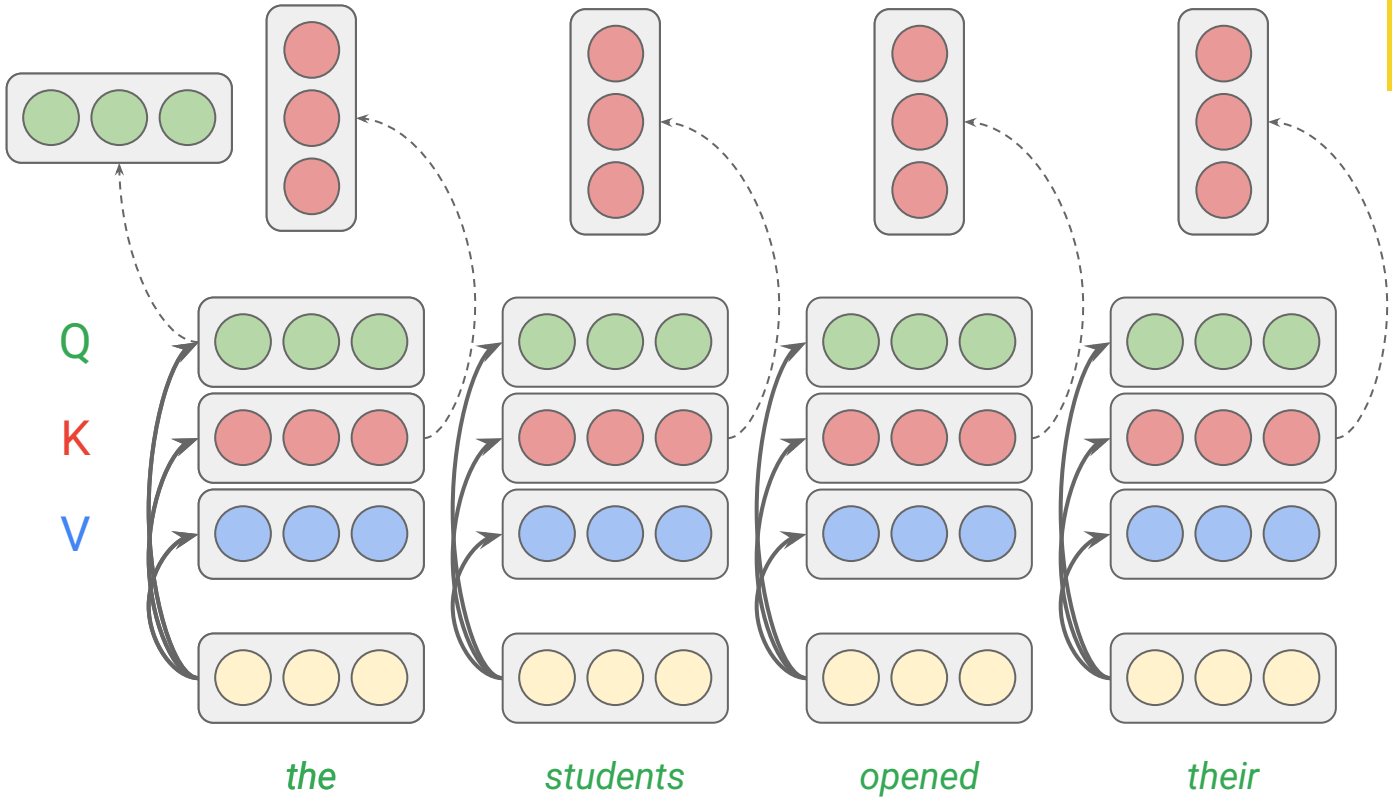
$$K = X \cdot W_K$$

$$V = X \cdot W_V$$

linear projections

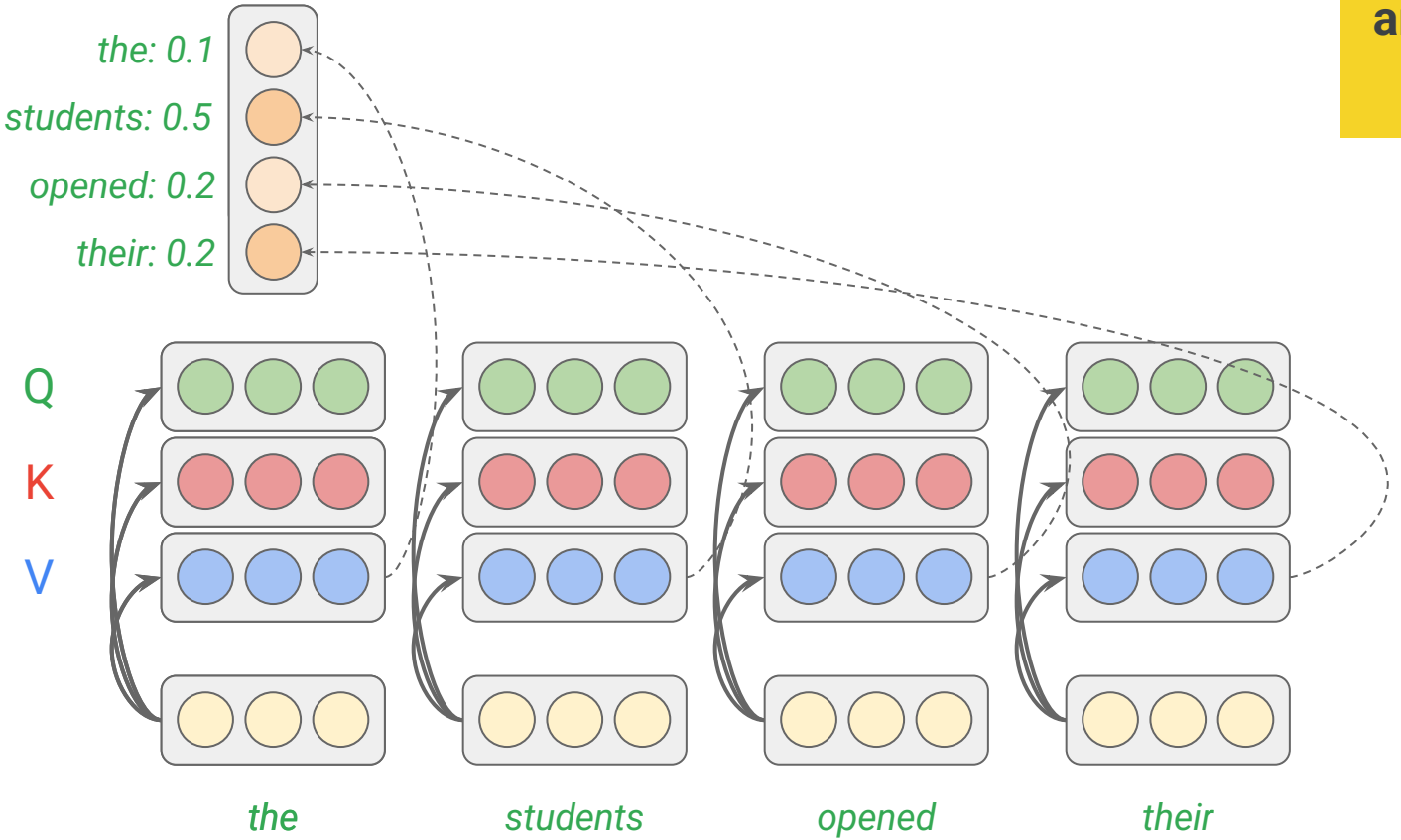
Self-attention (cont'd)

***all* computations are parallelized**



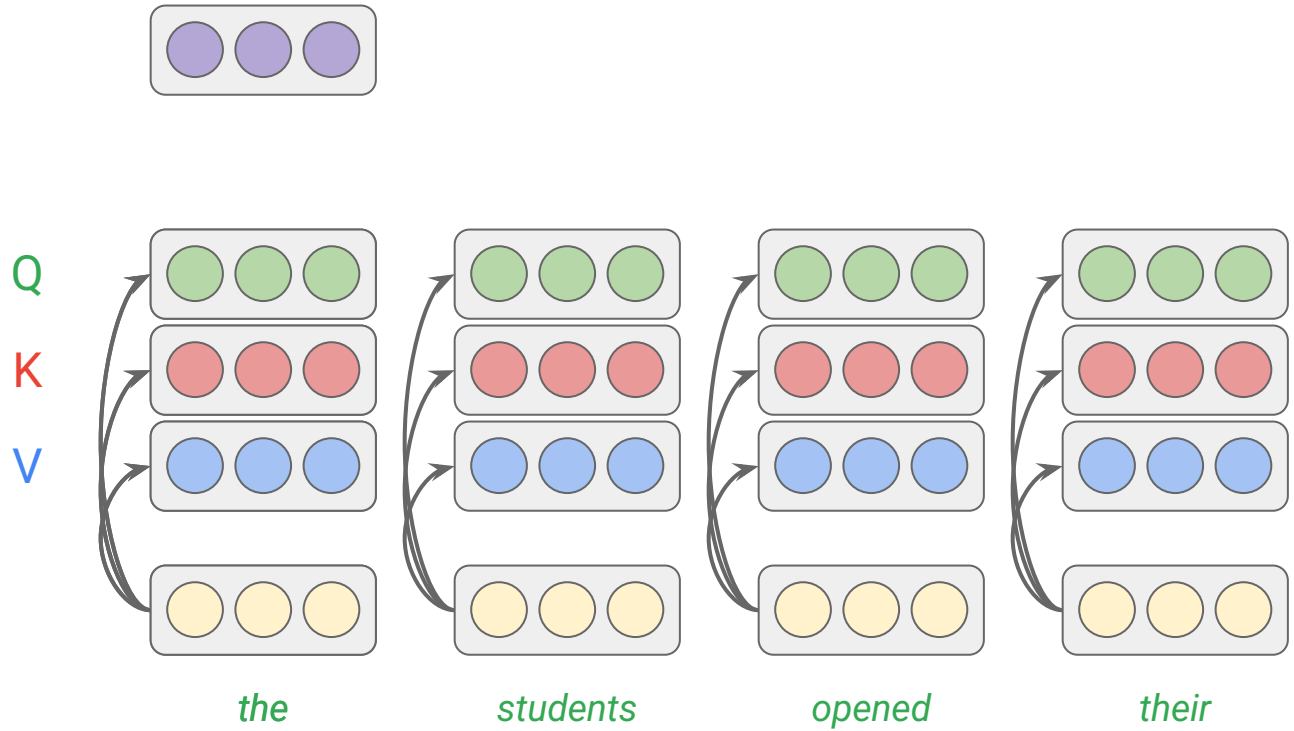
Self-attention (cont'd)

all computations are parallelized

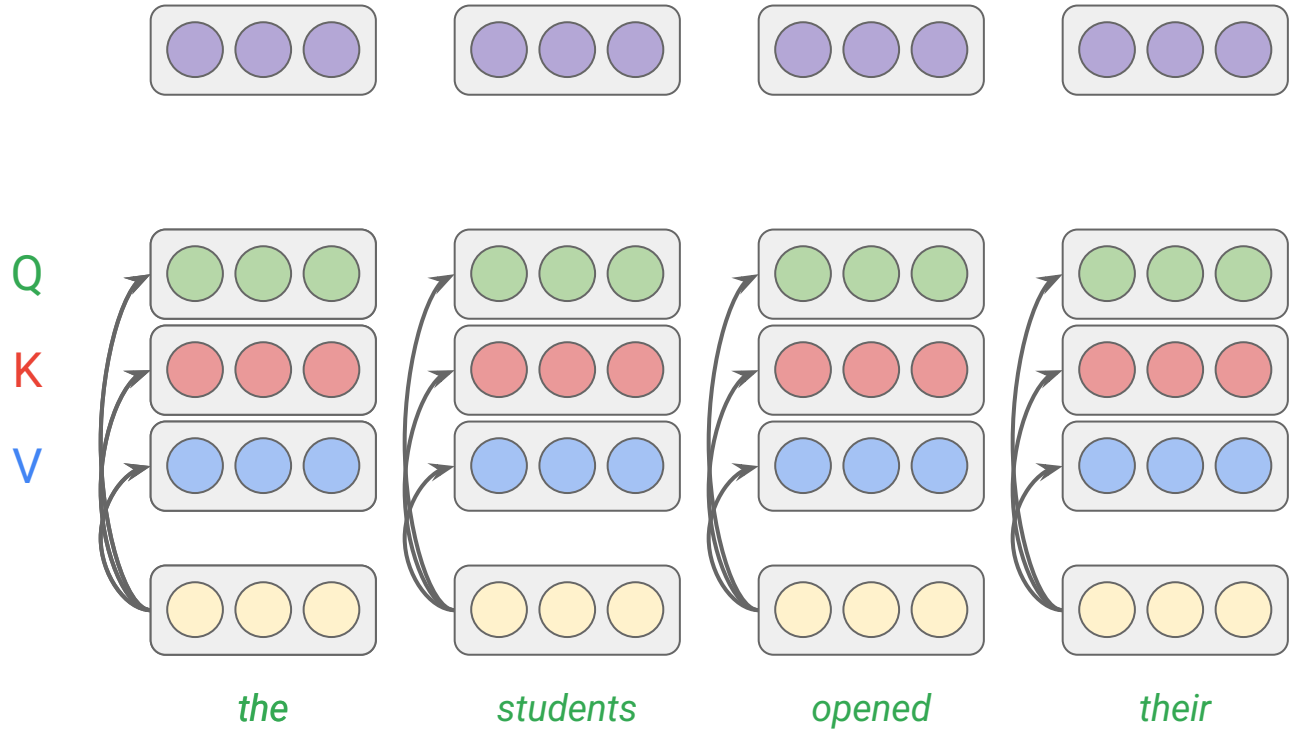


Self-attention (cont'd)

***all* computations are parallelized**

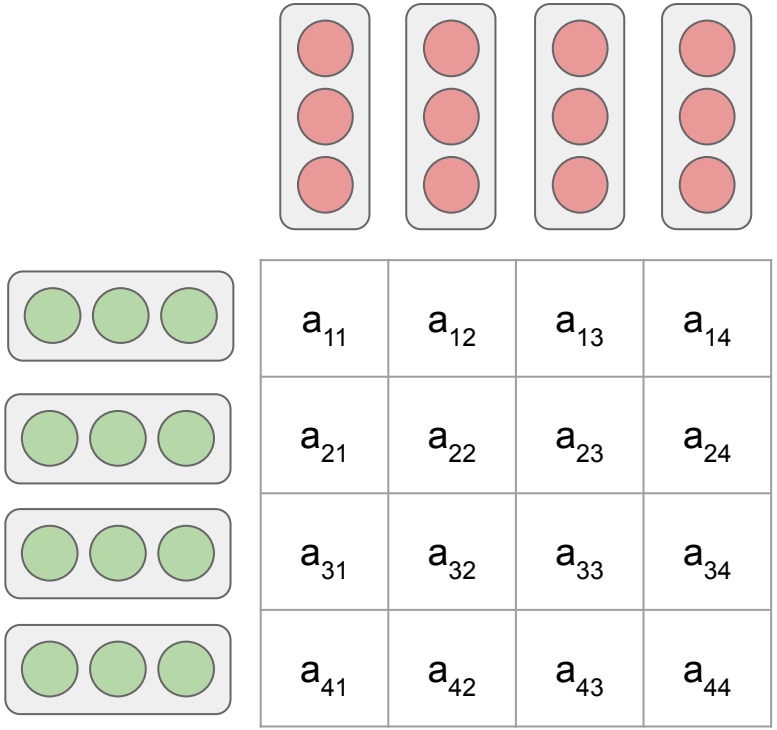


Self-attention (cont'd)



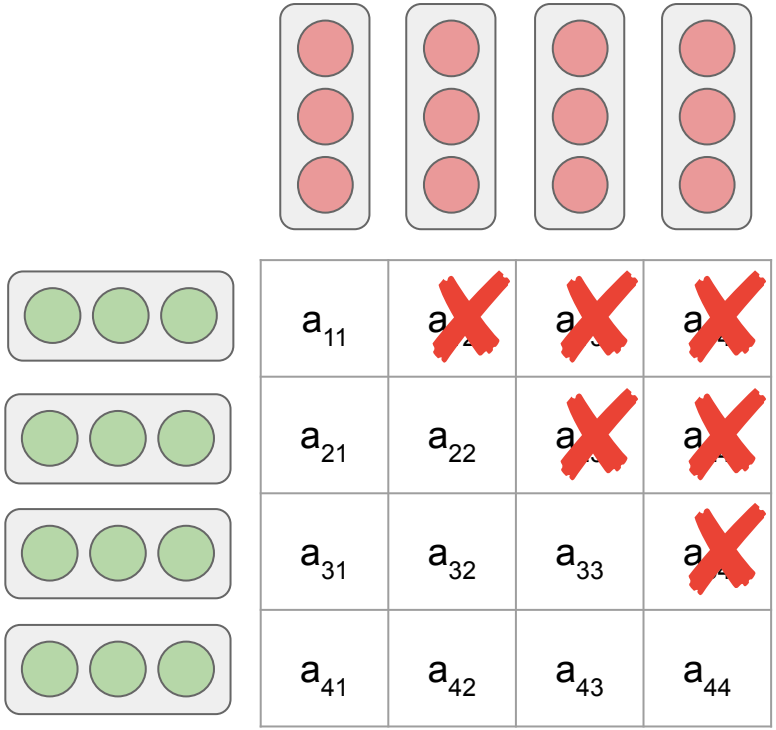
all **computations are parallelized during training and sequential during inference**

Quadratic complexity



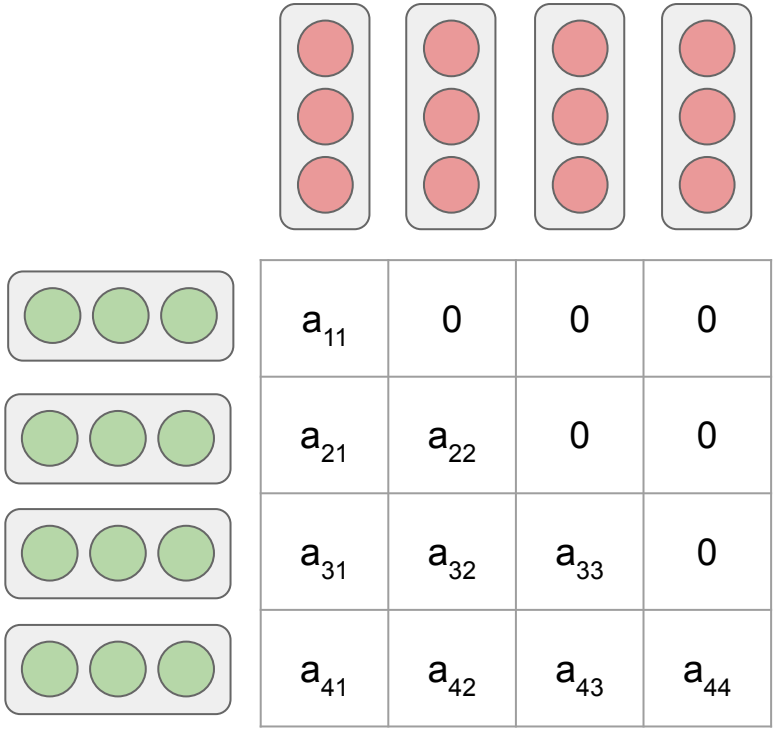
The time complexity of self-attention is quadratic in the input length $O(n^2)$

Self-attention in the decoder



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

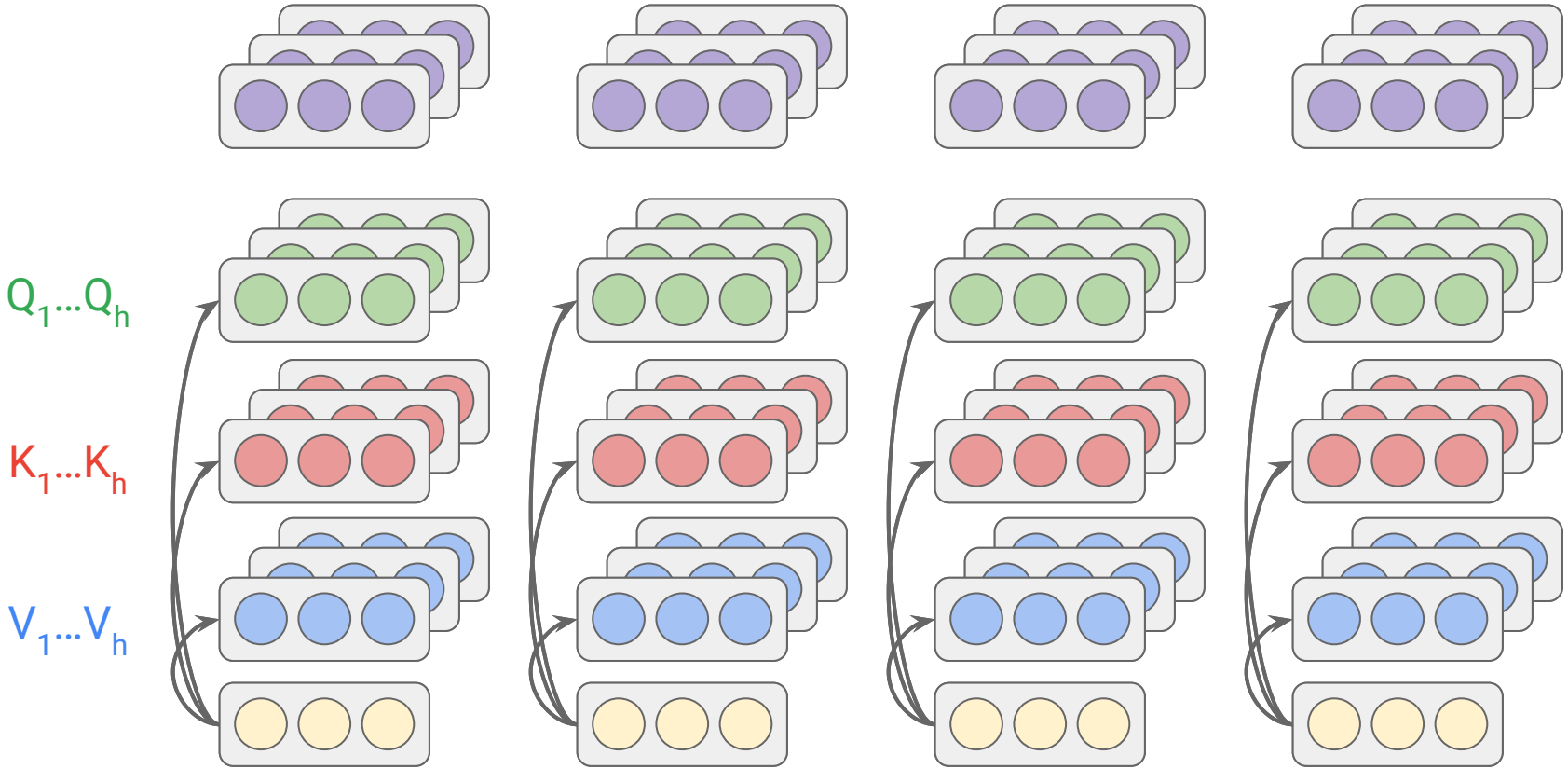
Self-attention in the decoder (cont'd)



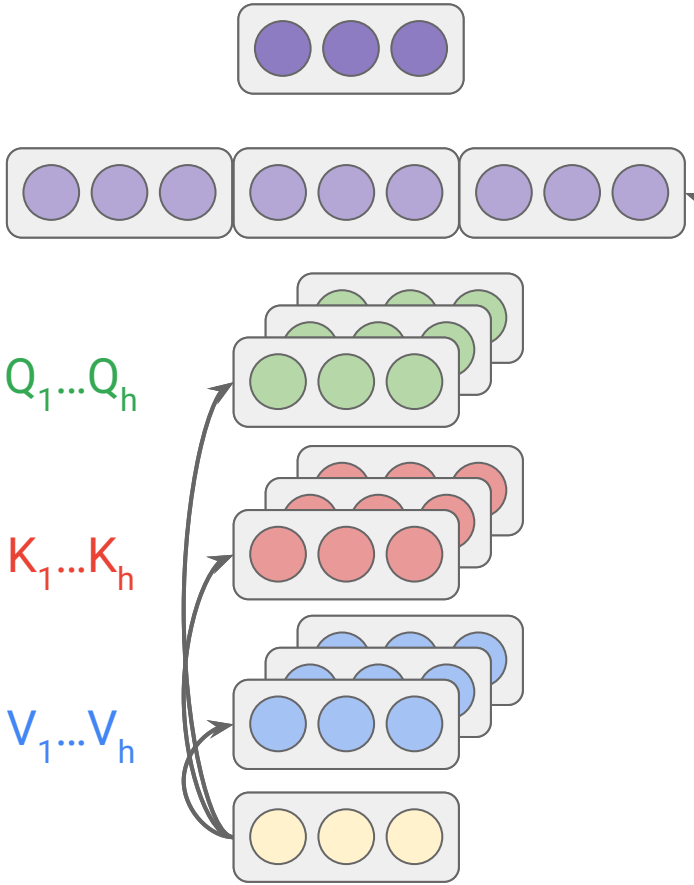
a_{11}	0	0	0
a_{21}	a_{22}	0	0
a_{31}	a_{32}	a_{33}	0
a_{41}	a_{42}	a_{43}	a_{44}

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

Multi-head attention

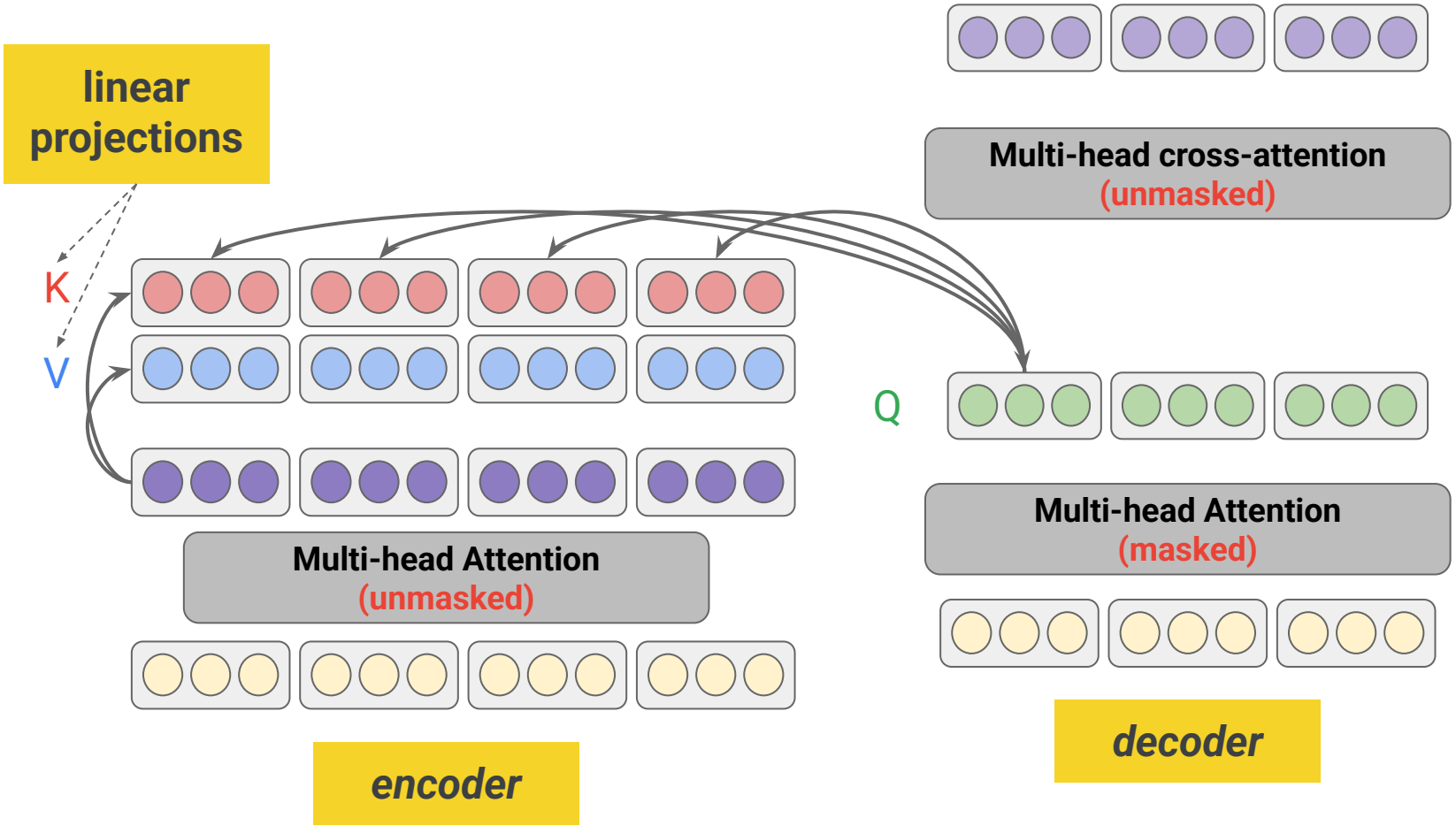


Multi-head attention (cont'd)

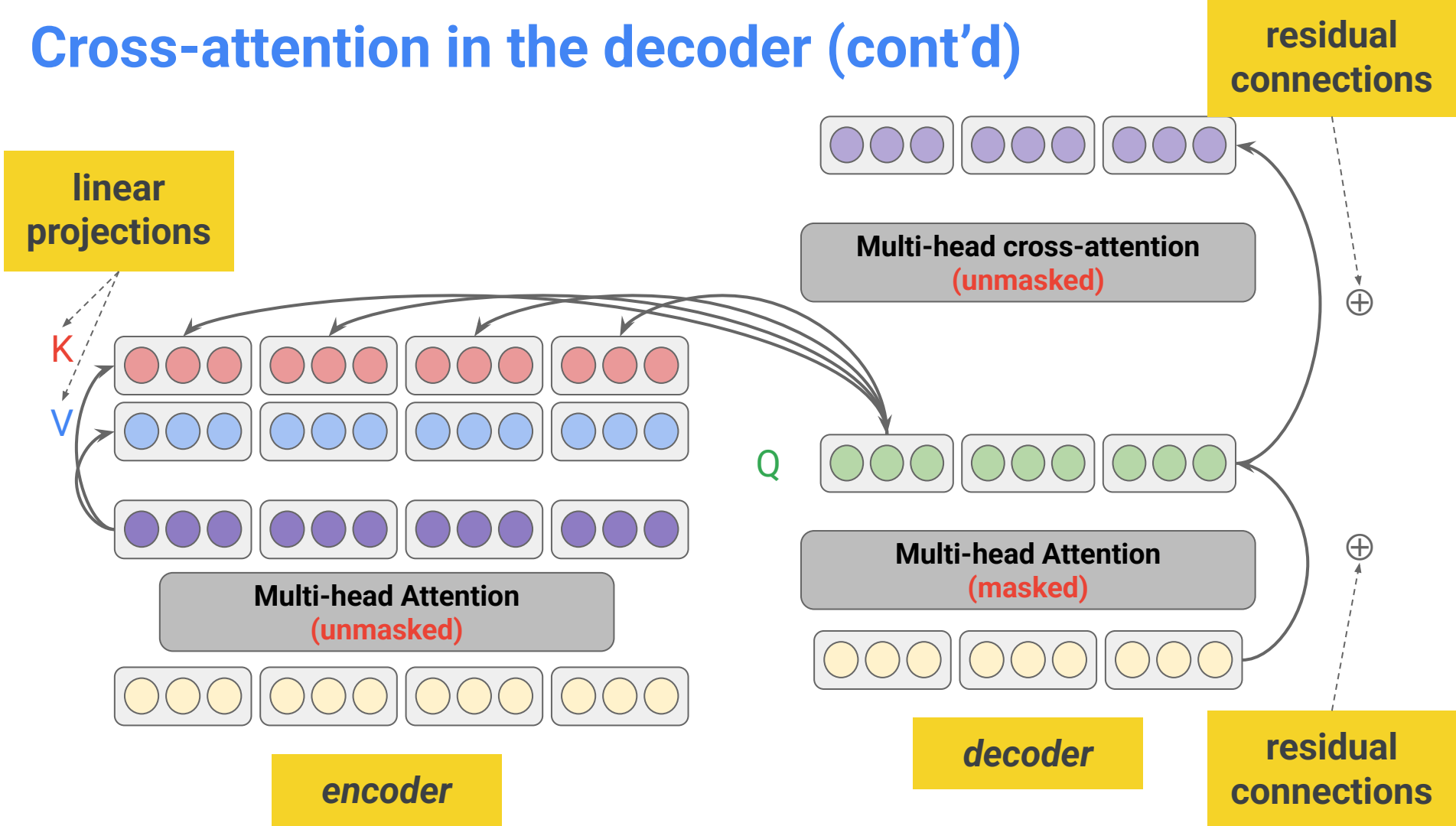


These output values are concatenated and once again projected

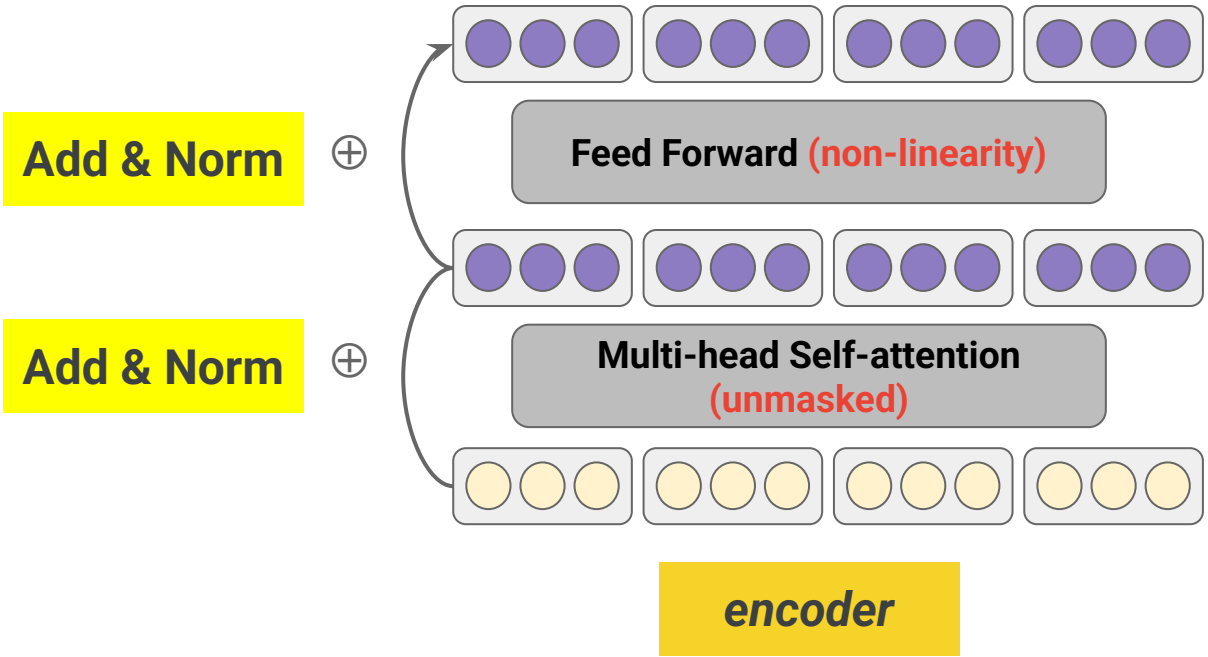
Cross-attention in the decoder



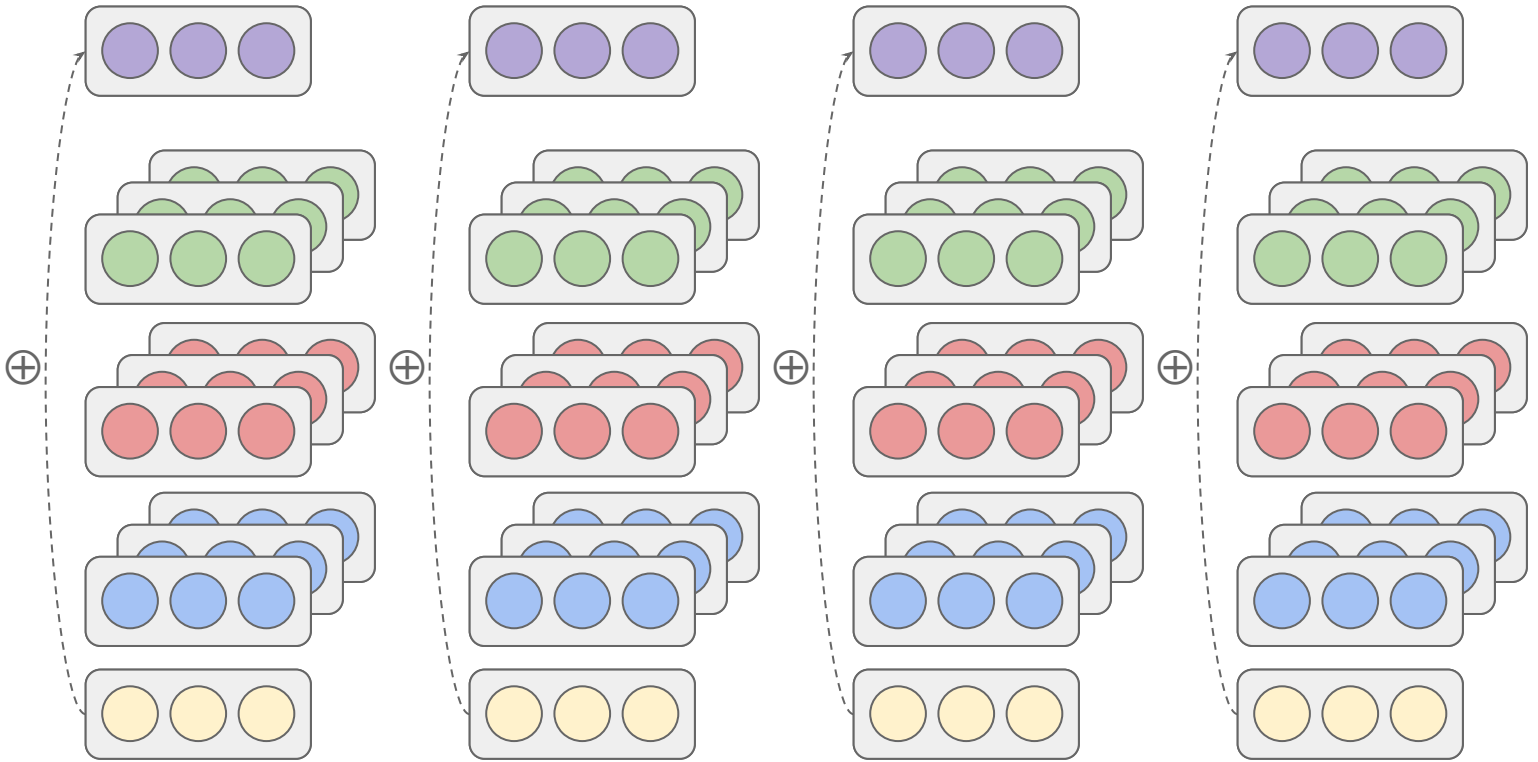
Cross-attention in the decoder (cont'd)



Encoder (one layer)



Residual connection



Residual connection

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

Layer normalization

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

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

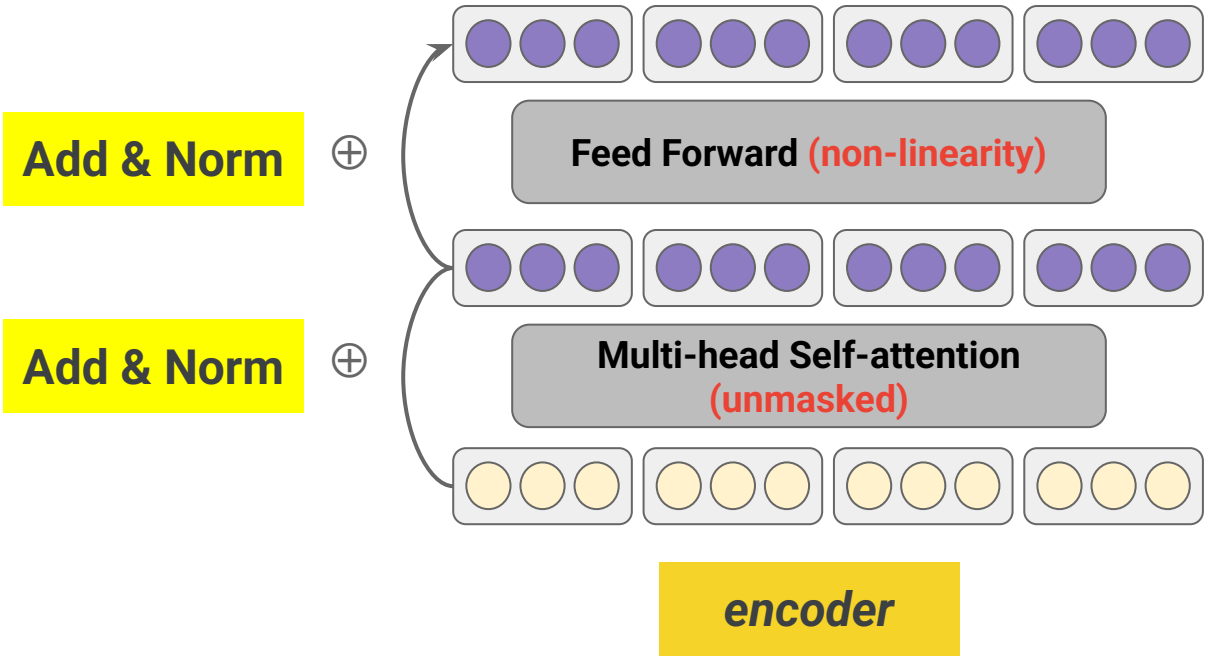
$$\text{LayerNorm}(x + \text{Sublayer}(x))$$

Position-wise Feed-Forward Networks

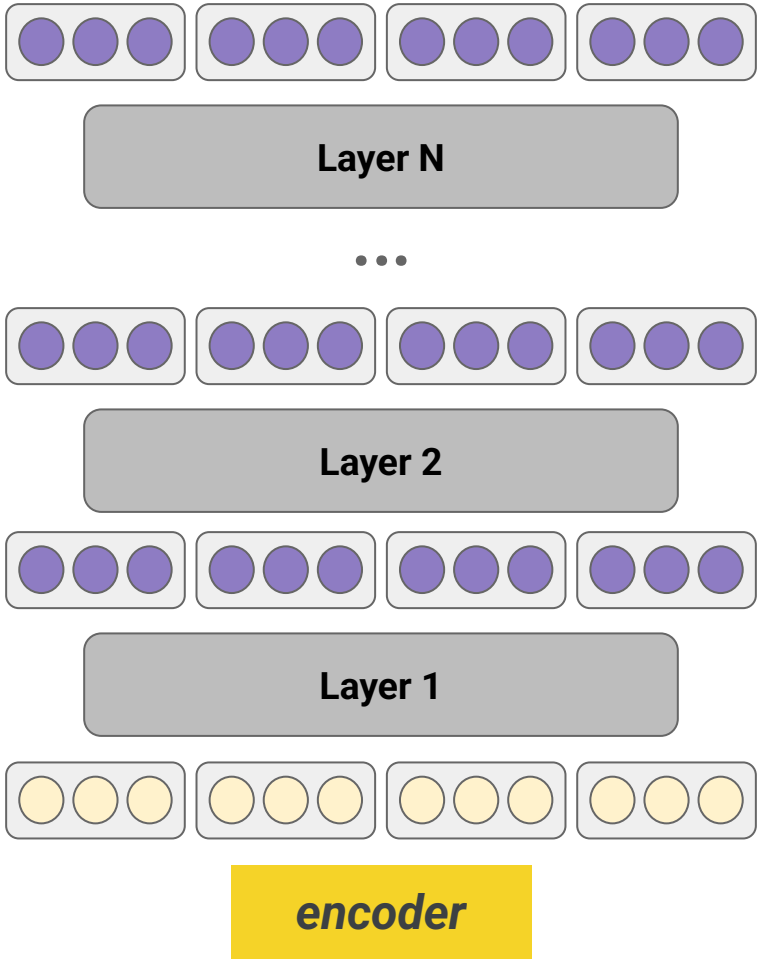
$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

ReLU (Rectified Linear Unit)

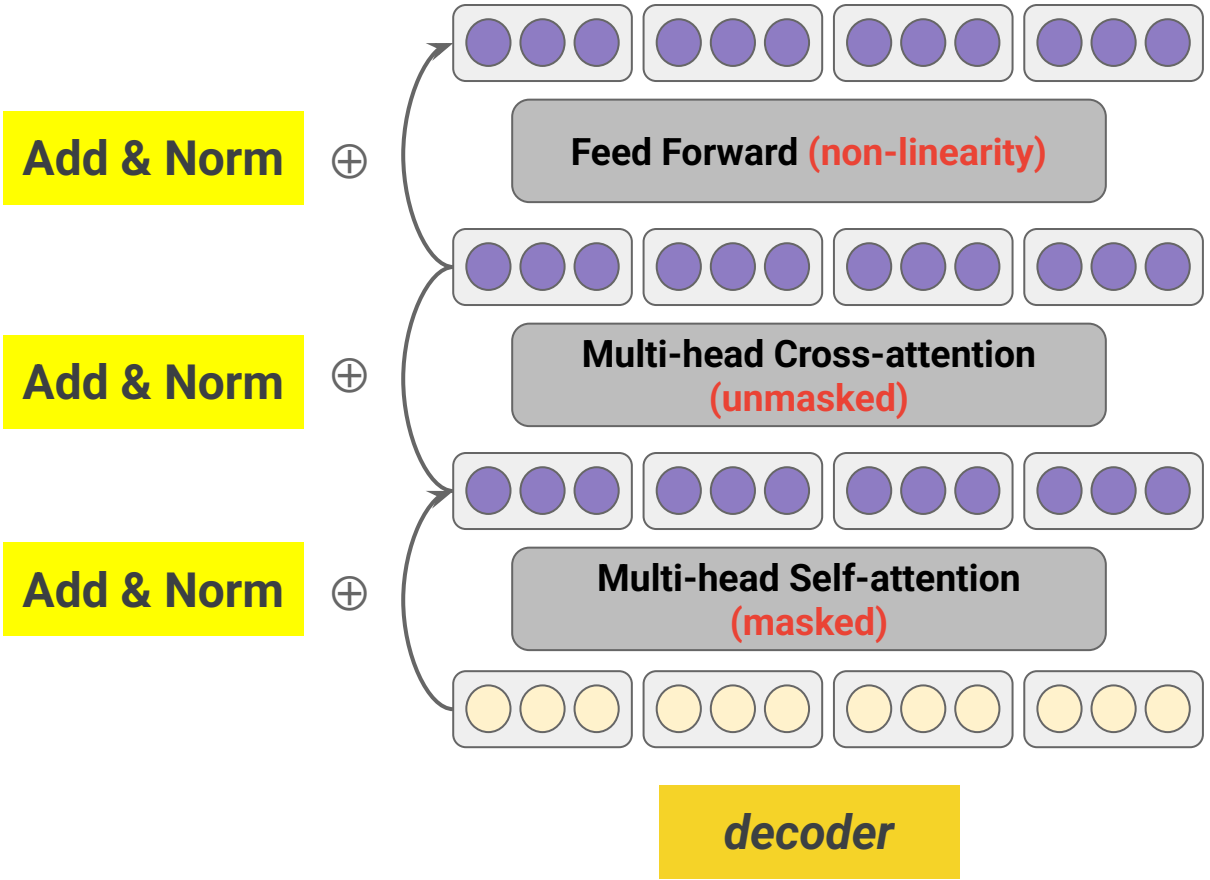
Encoder (one layer)



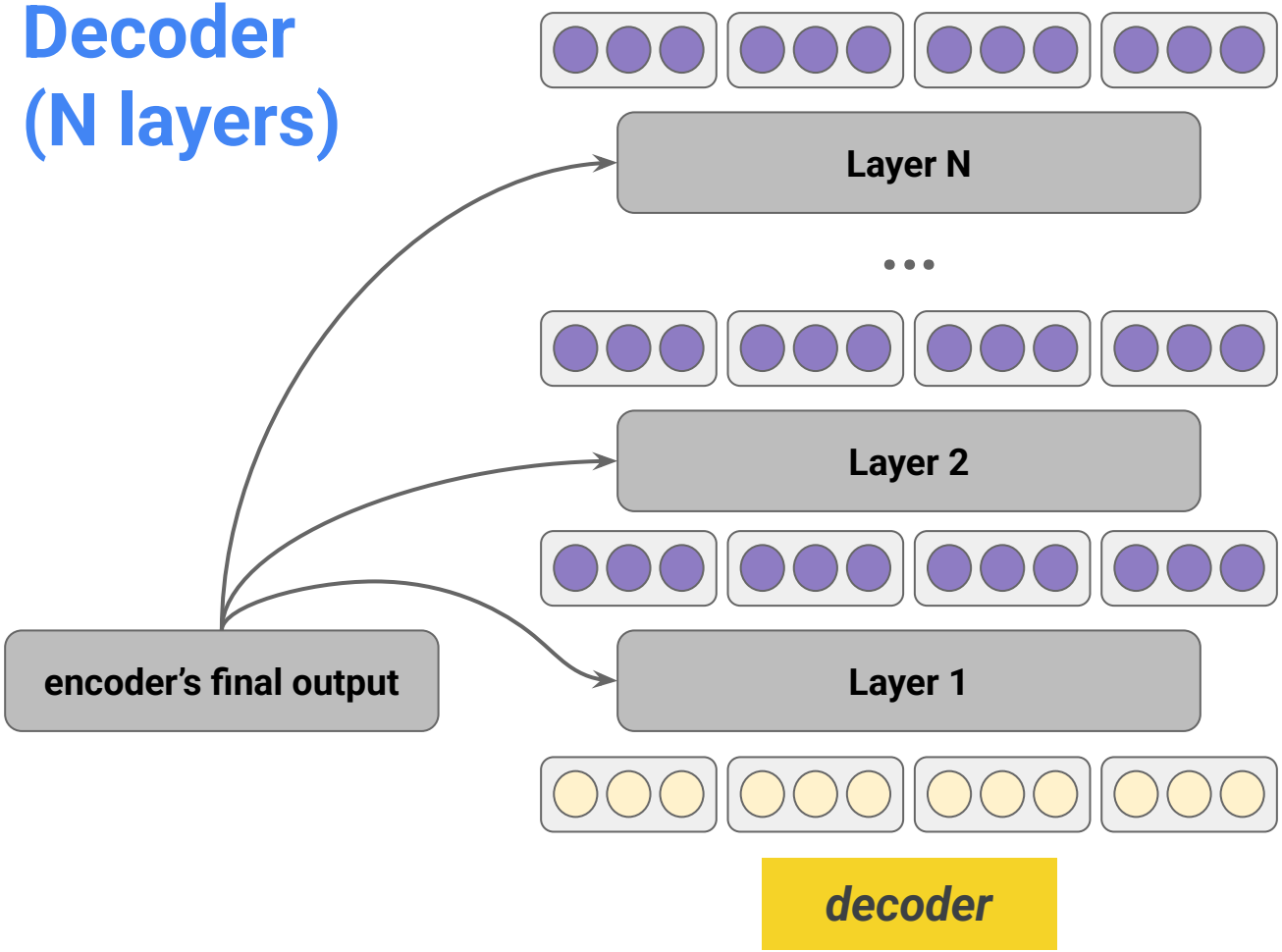
Encoder (N layers)



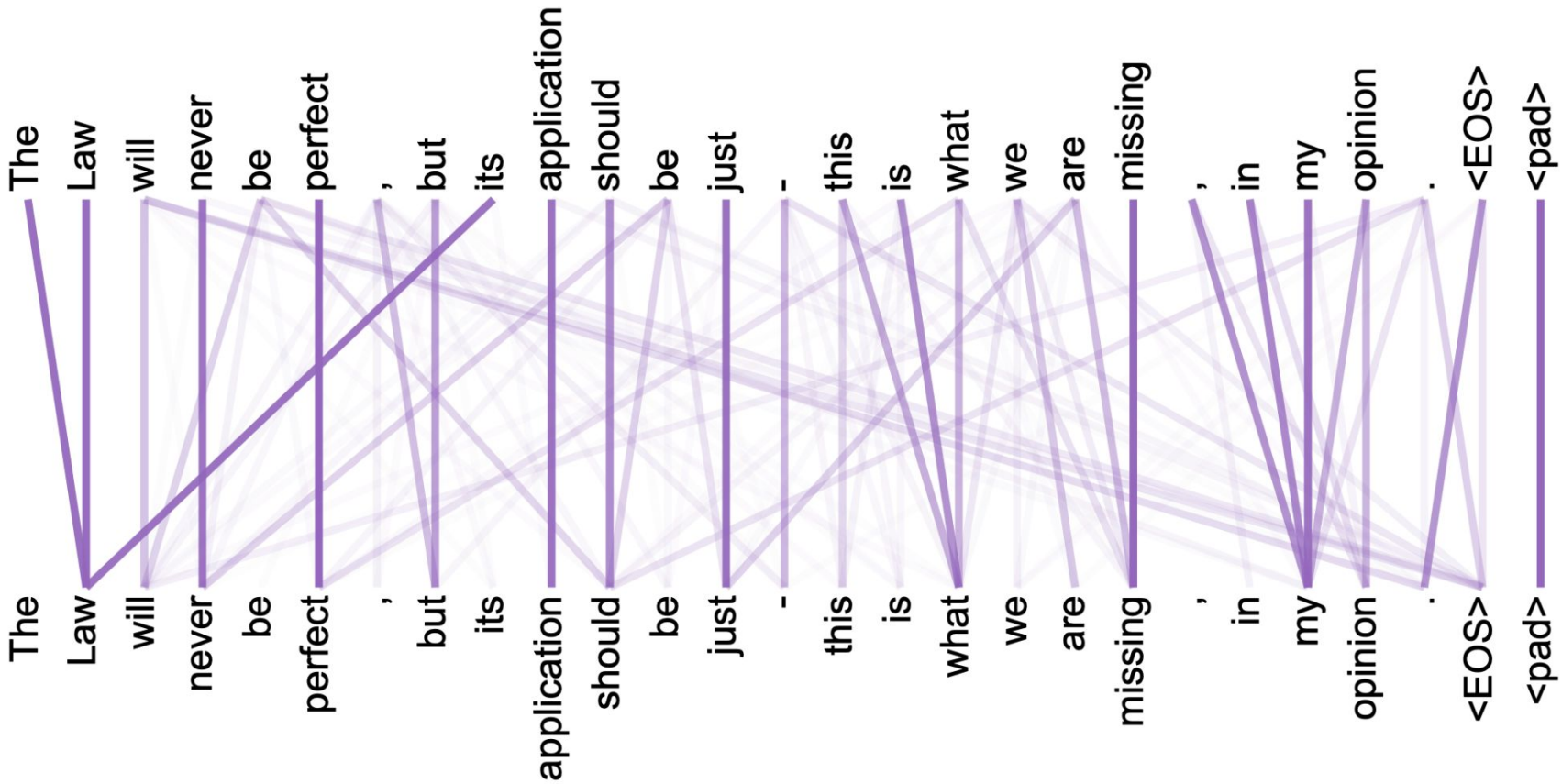
Decoder (one layer)



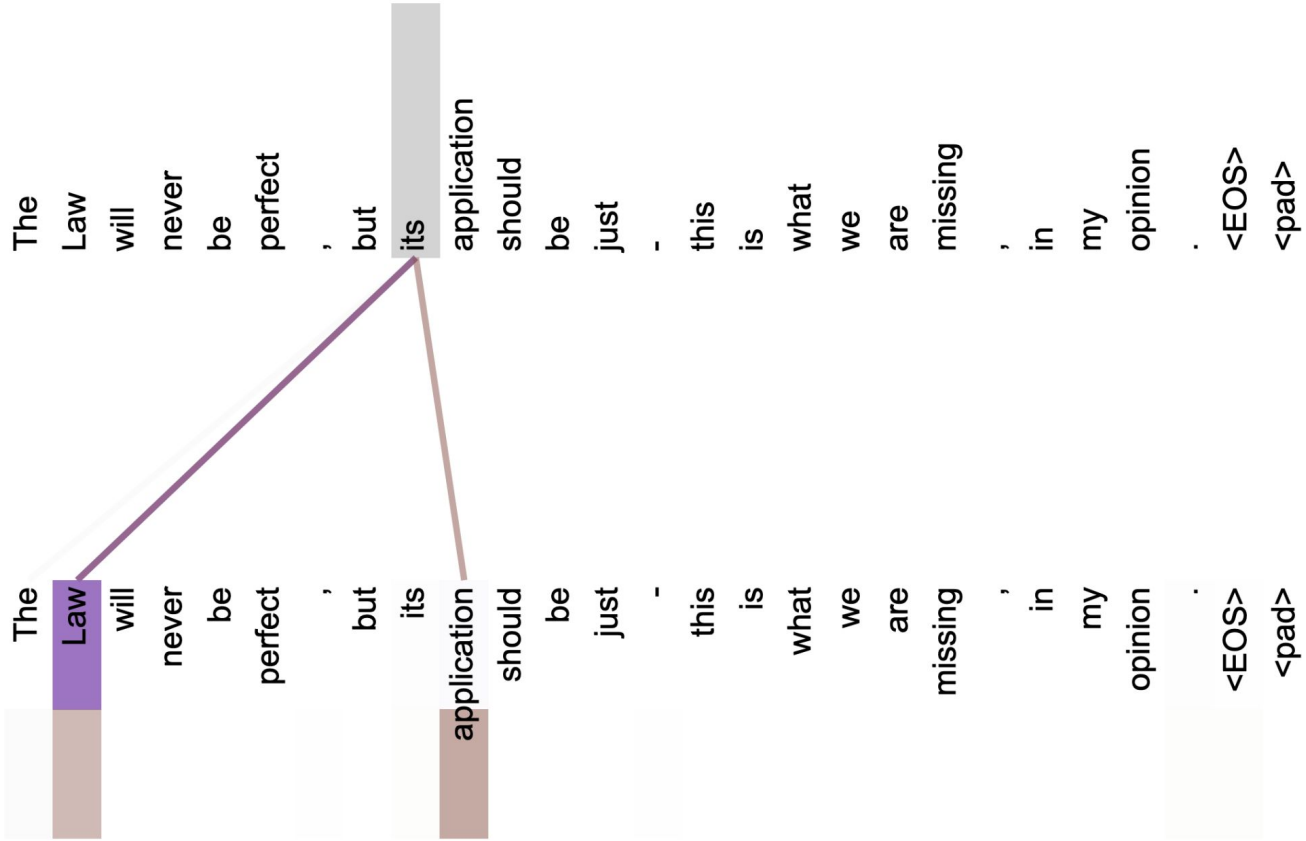
Decoder (N layers)



Attention visualizations



Attention visualizations (cont'd)

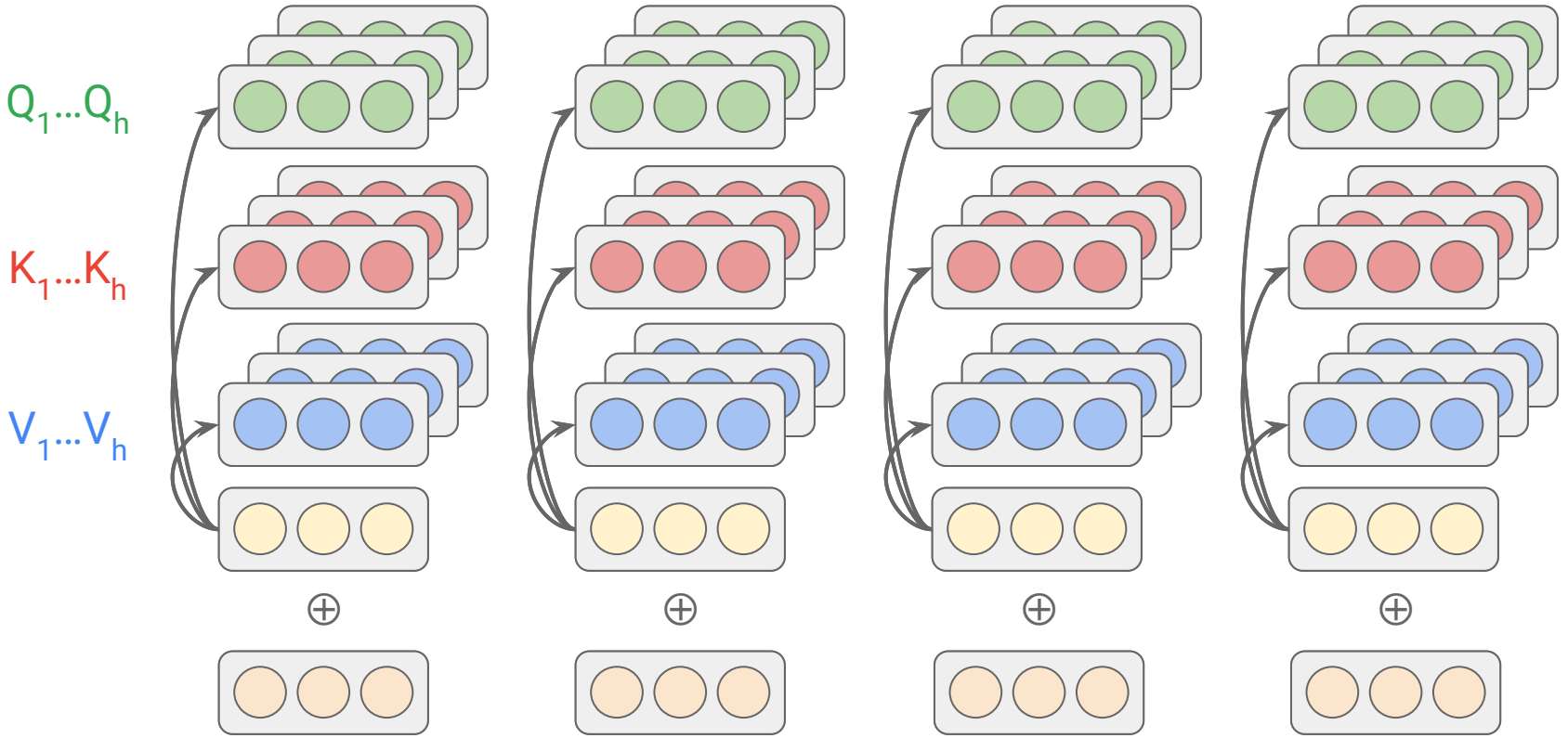


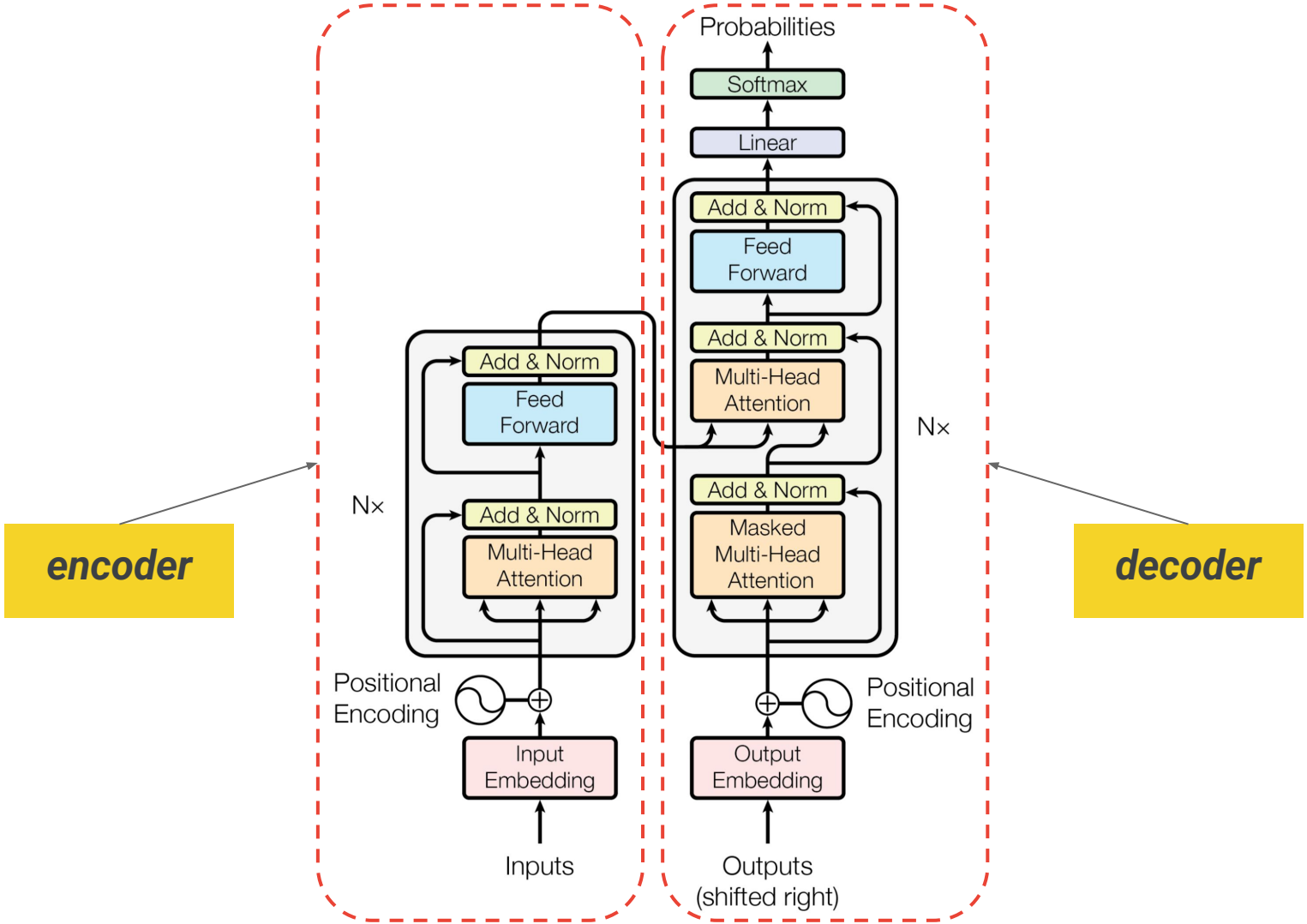
Sinusoidal positional encoding

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

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

Positional Encoding (cont'd)




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

Different model architectures

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

Image created by Gemini

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

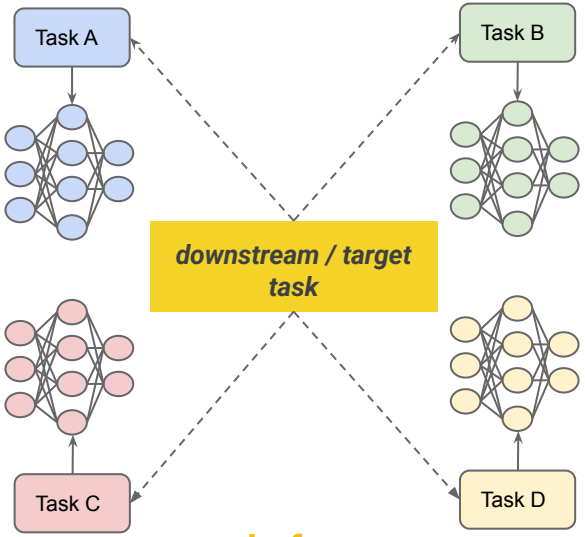
Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**

Google AI Language

{jacobdevlin,mingweichang,kentonl,kristout}@google.com

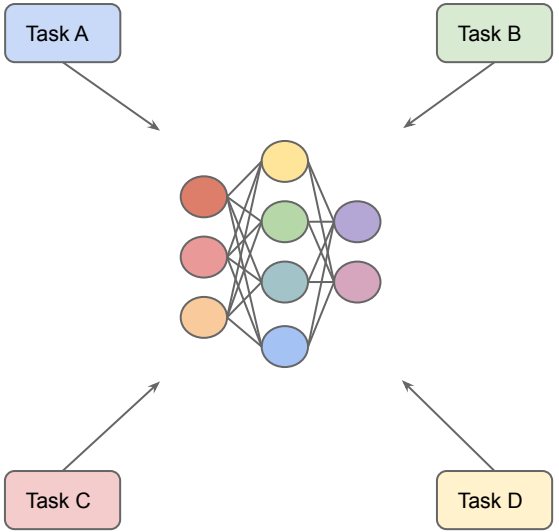
A learning paradigm shift

training task-specific models from scratch



before BERT

pretraining and then adapting



since BERT

ELMo

Image created by Gemini

Deep contextualized word representations

Matthew E. Peters[†], **Mark Neumann**[†], **Mohit Iyyer**[†], **Matt Gardner**[†],
{matthewp,markn,mohiti,mattg}@allenai.org

Christopher Clark[*], **Kenton Lee**[*], **Luke Zettlemoyer**[†*]
{csquared,kentonl,lsz}@cs.washington.edu

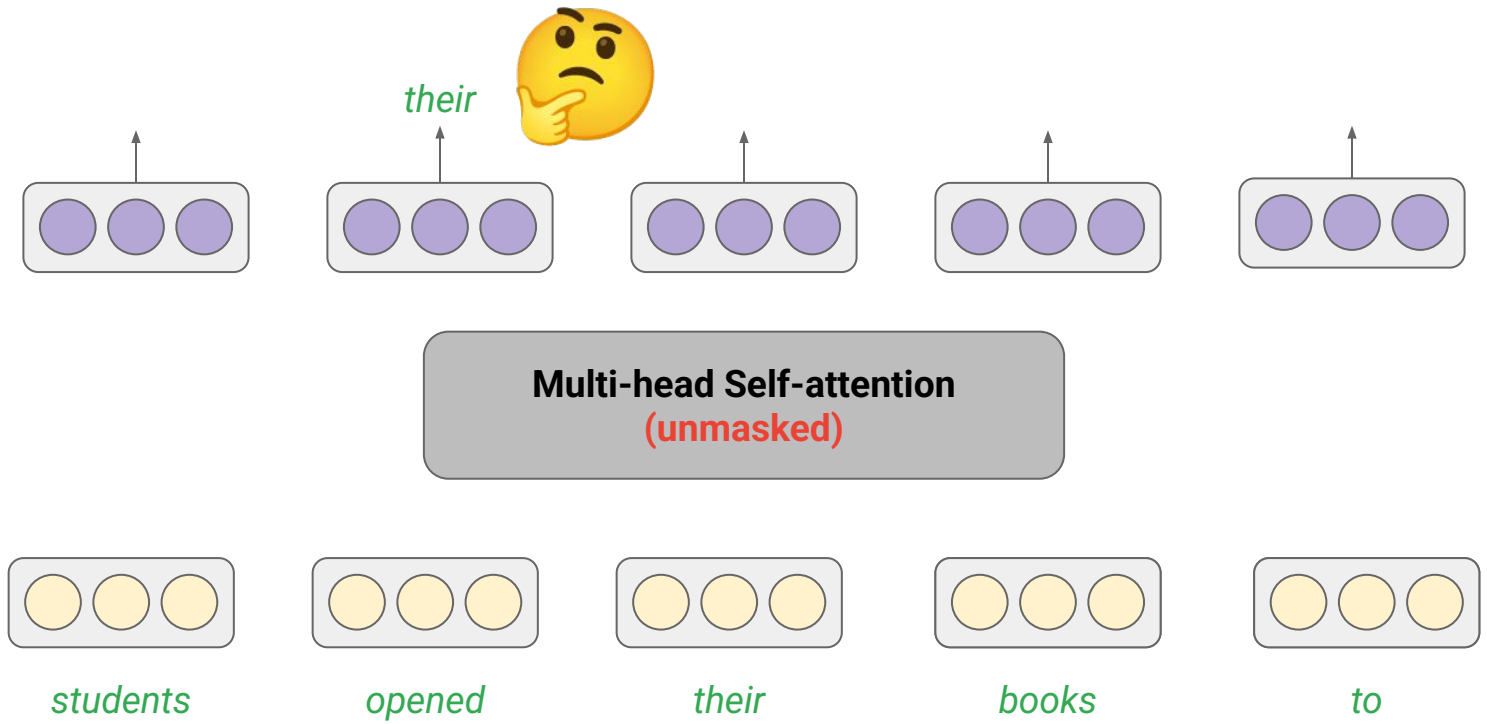
[†]Allen Institute for Artificial Intelligence
[*]Paul G. Allen School of Computer Science & Engineering, University of Washington

BERT vs. ELMo

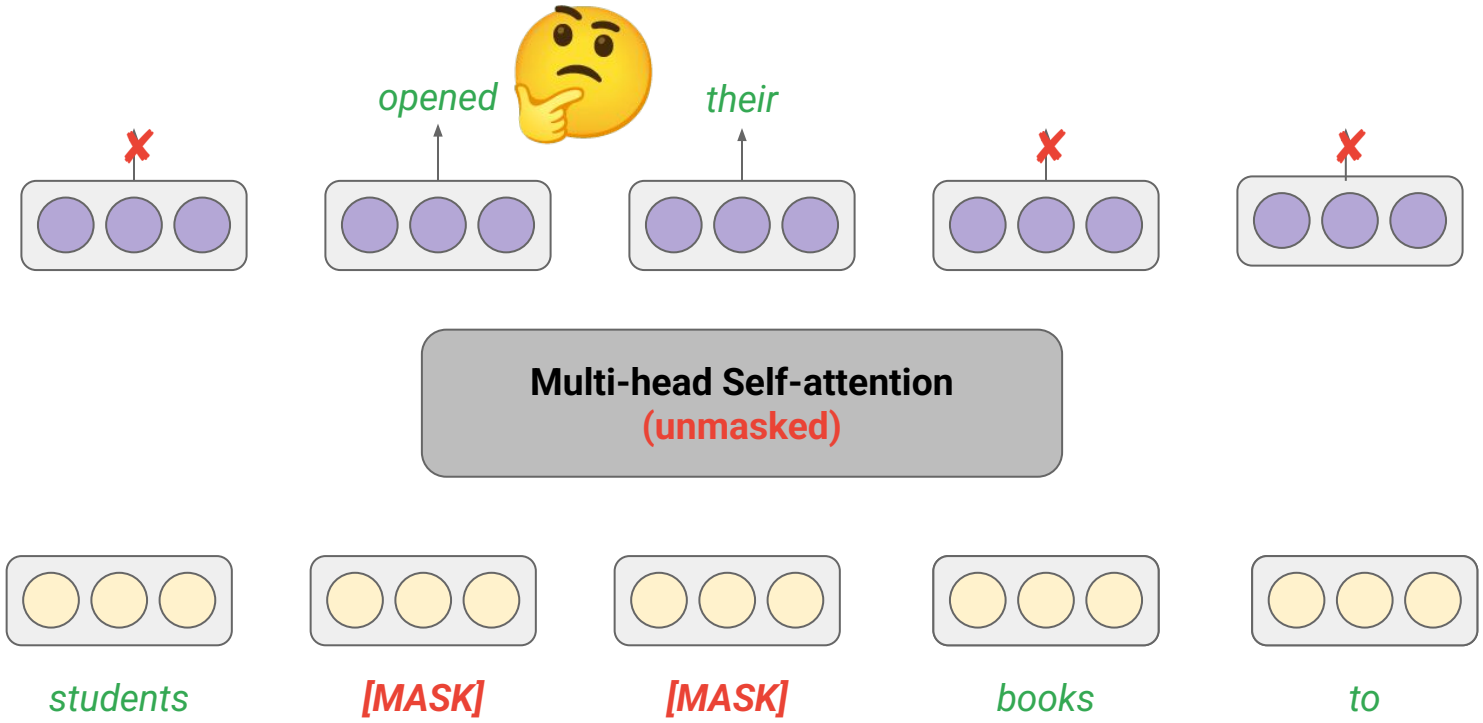
	BERT	ELMo
Model	Transformers	Bidirectional LSTM (Long Short-Term Memory, a variant of RNN)
Pre-training objective(s)	Masked language modeling + next sentence prediction	Left-to-right language modeling
Adaptation method	Fine-tuning	Feature-based (pretrained representations as additional features to task-specific models)

Pretraining

Language modeling using a Transformer encoder

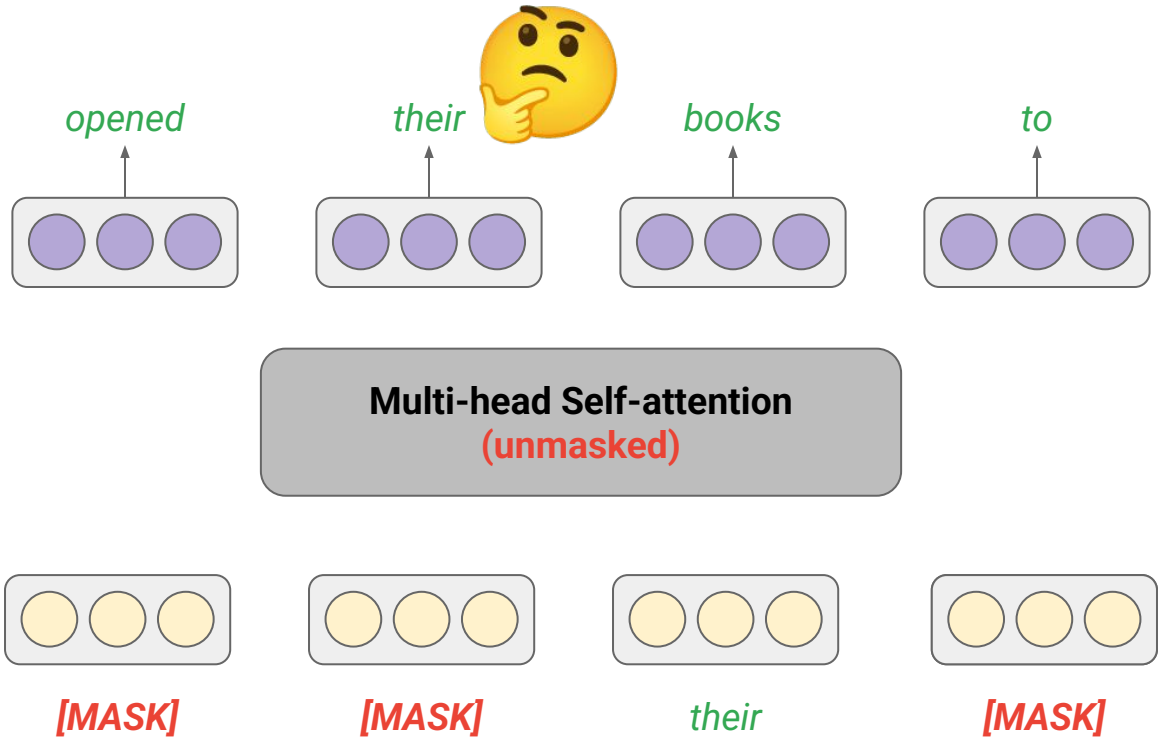


Masked language modeling



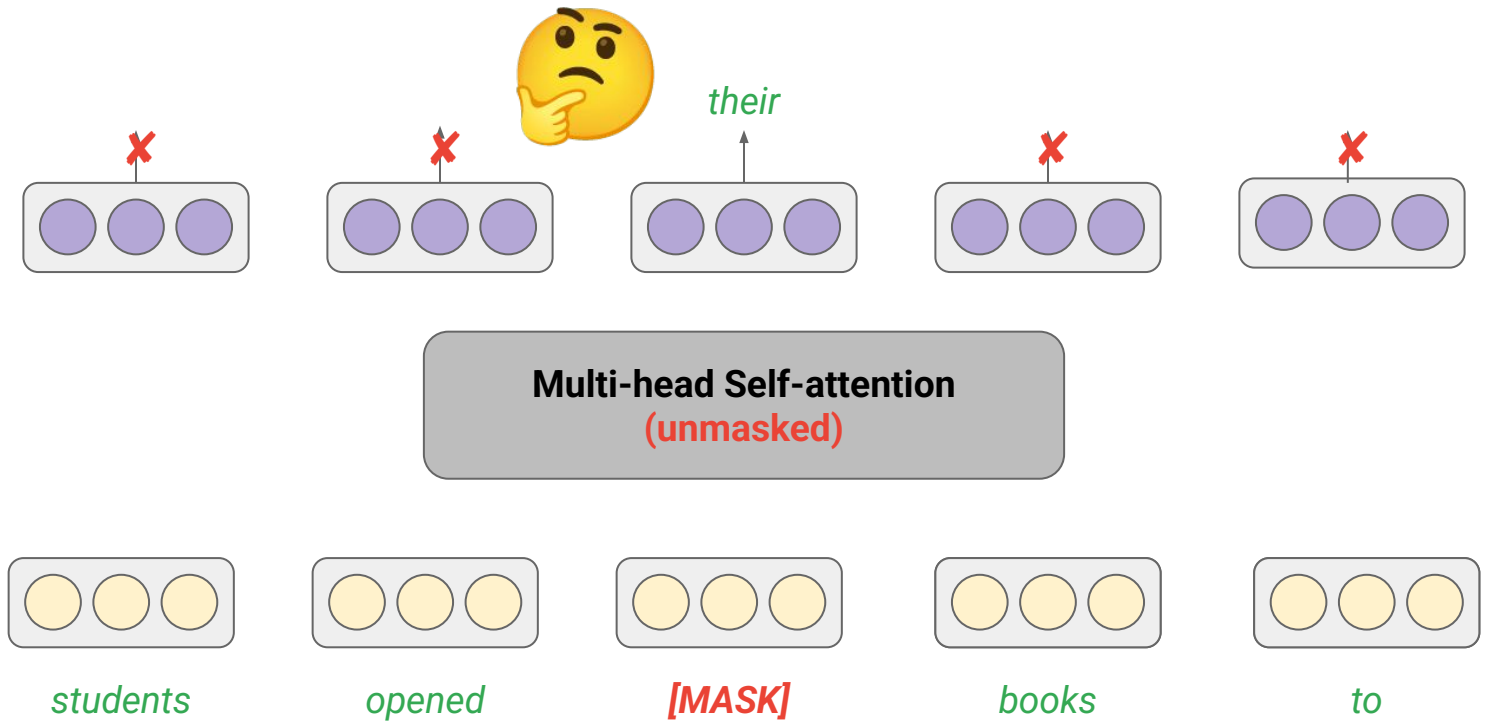
15% - 30% of all tokens in each sequence are masked at random

What if we mask more tokens?



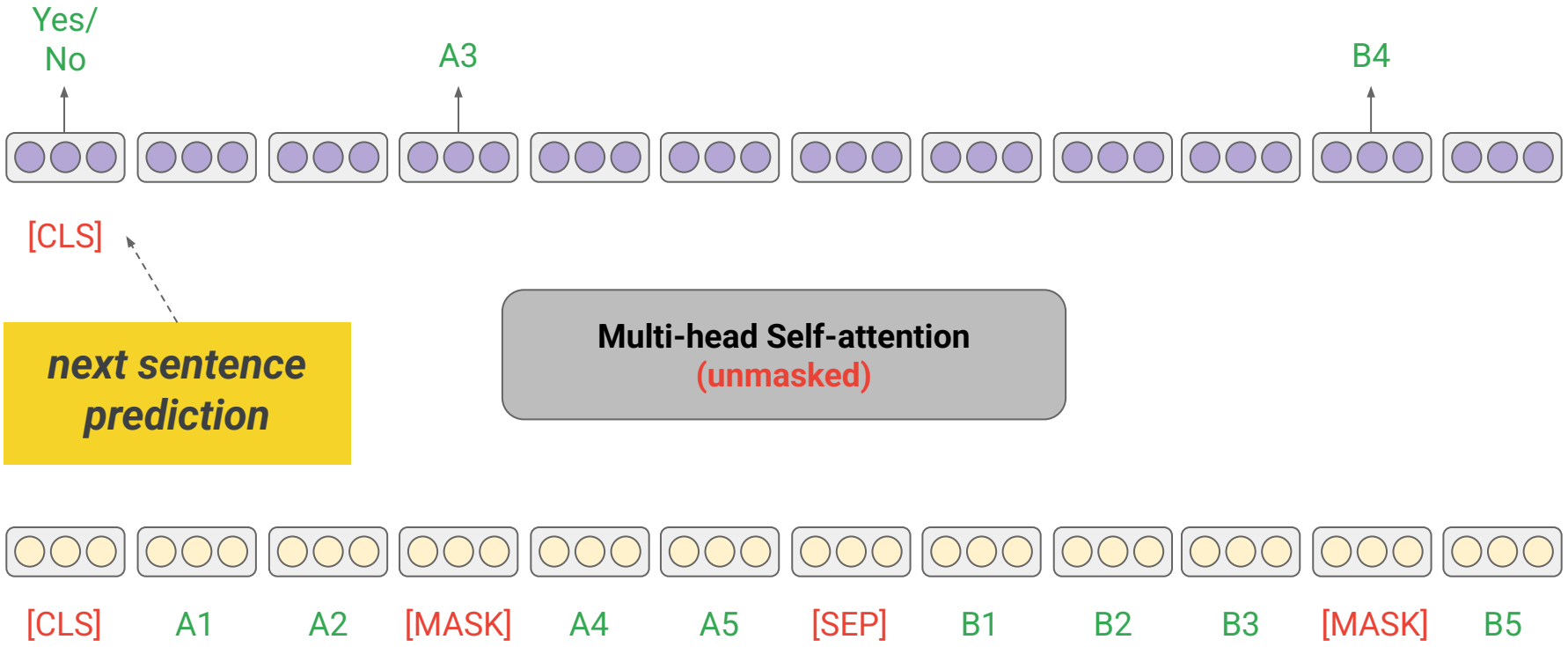
15% - 30% of all tokens in each sequence are masked at random

What if we mask less tokens?

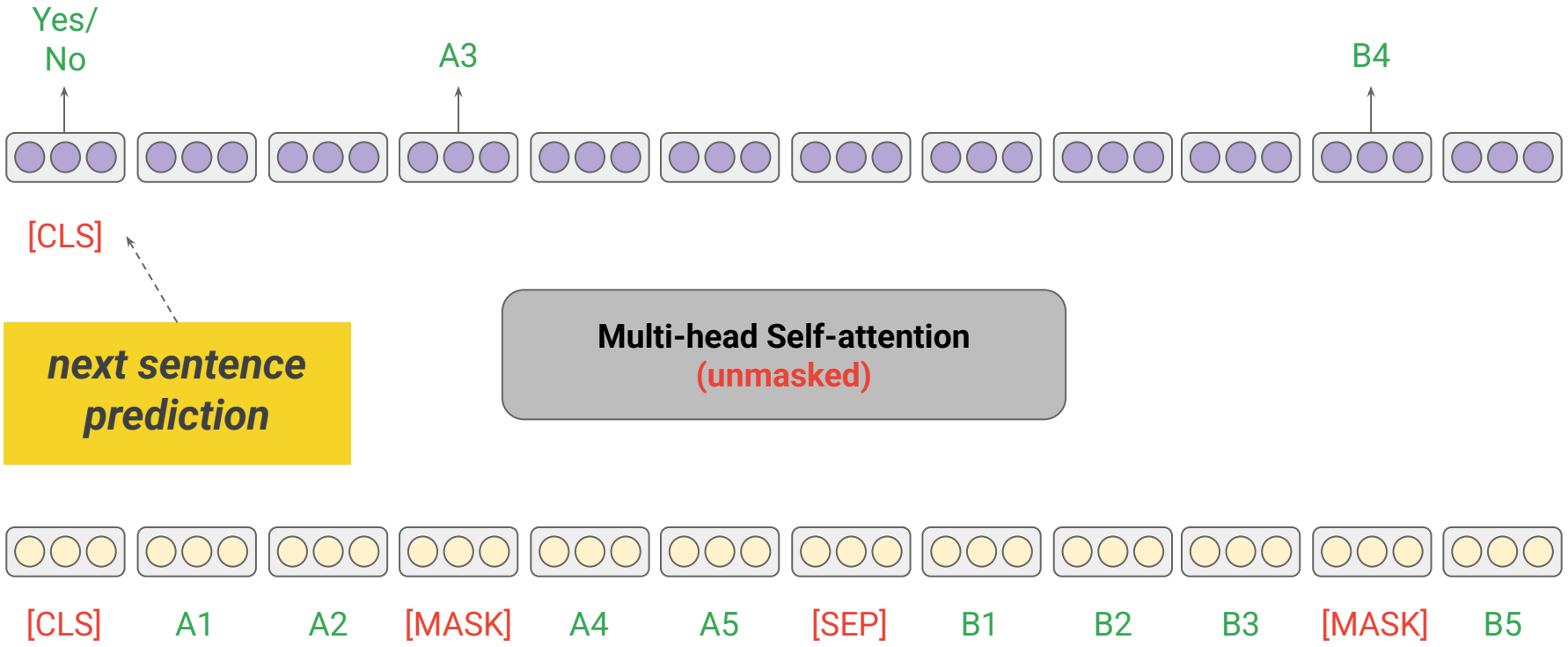


15% - 30% of all tokens in each sequence are masked at random

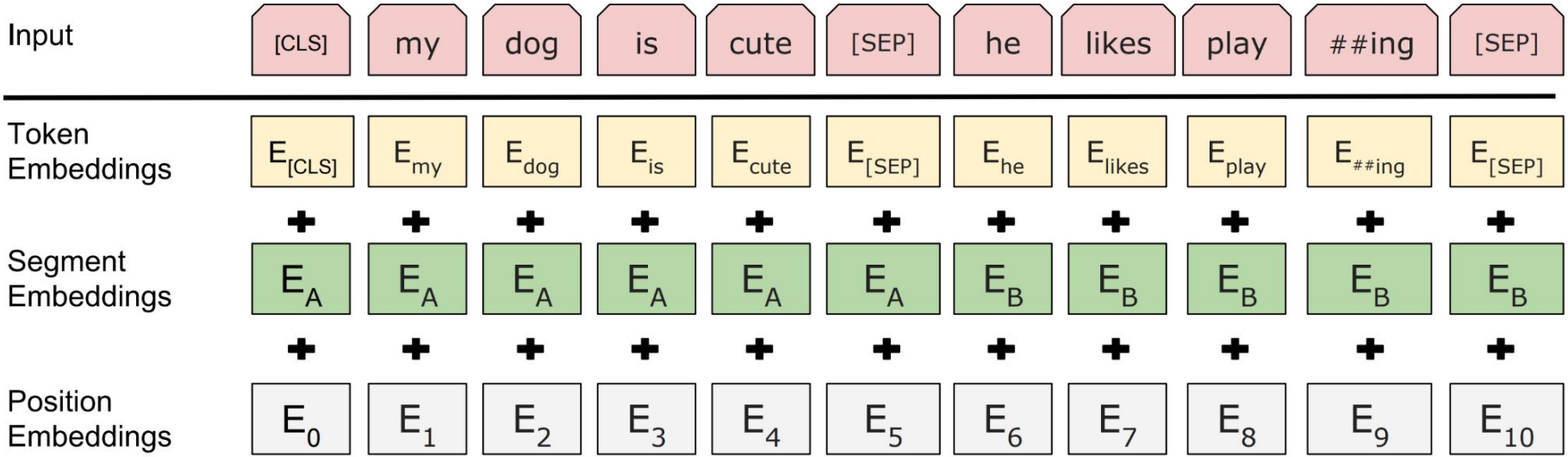
CLS & SEP tokens



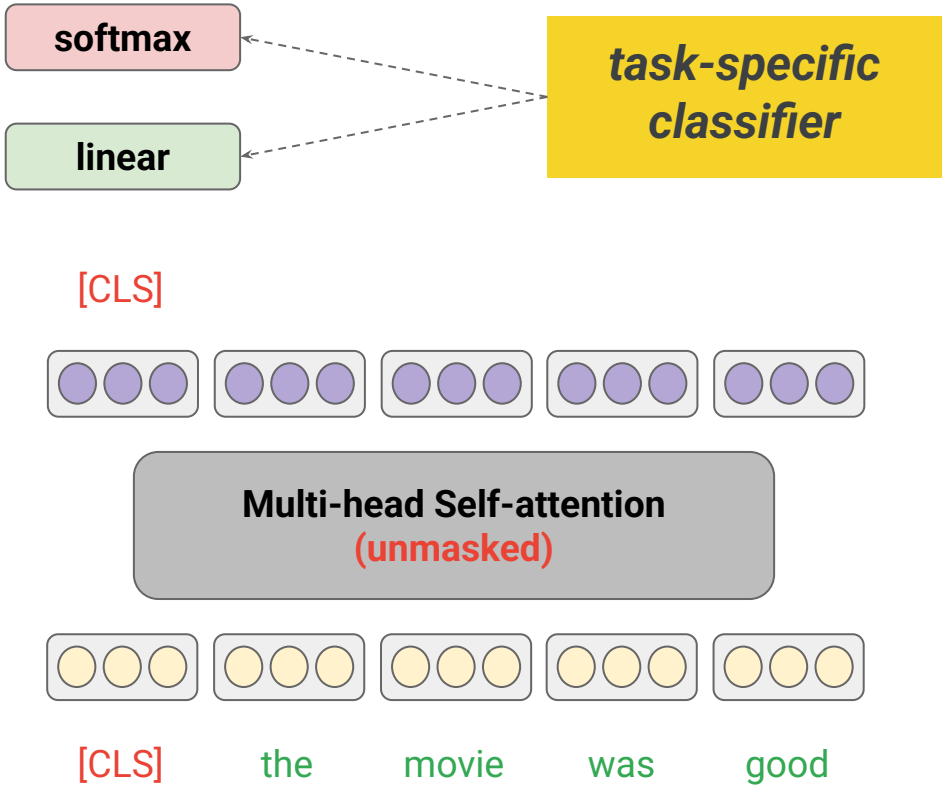
BERT Pretraining



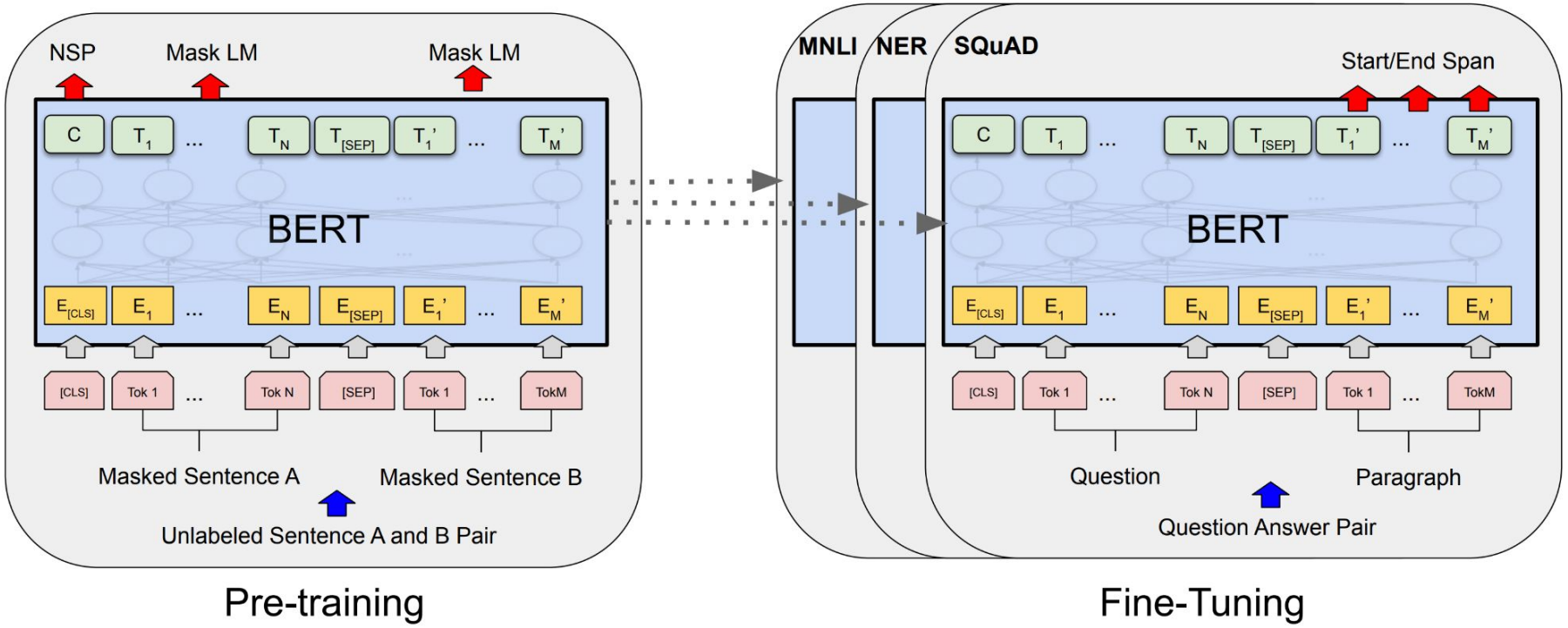
BERT input representation



BERT Fine-tuning

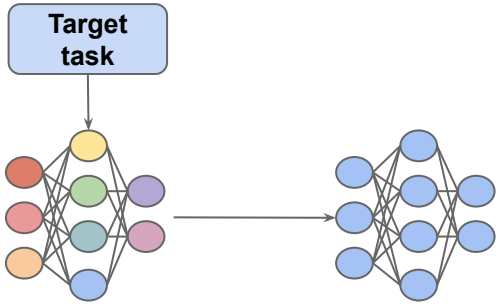


BERT Pretraining & Fine-tuning



Intermediate-task transfer / fine-tuning

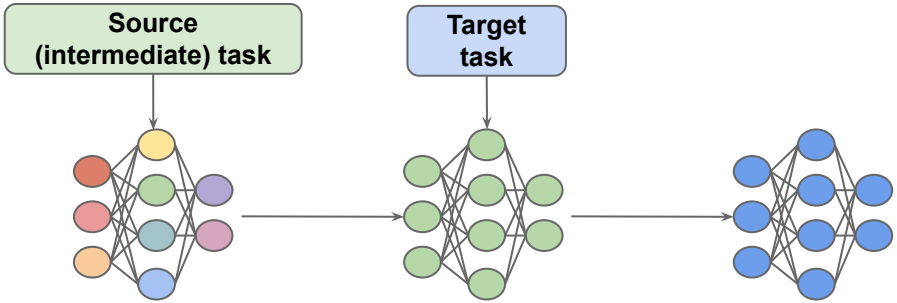
Standard Fine-tuning



BERTBASE → **RTE**: 63.5 ± 2.3

BERTLARGE → **RTE**: 68.6 ± 7.2

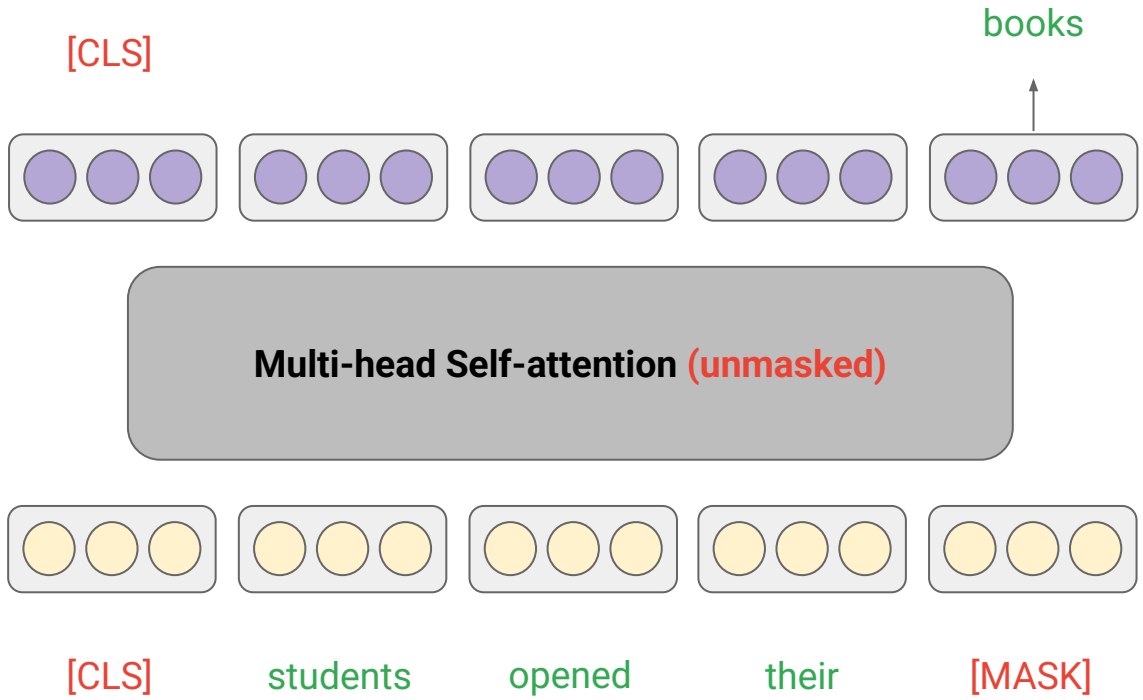
Intermediate-task transfer



BERTBASE → **MNLI** → **RTE**: **78.1 ± 1.9**

BERTLARGE → **MNLI** → **RTE**: **82.3 ± 1.4**

Can BERT be used for text generation?



iterative masking and unmasking

T5: Text-to-Text Transfer Transformer

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Colin Raffel* CRAFFEL@GMAIL.COM

Noam Shazeer* NOAM@GOOGLE.COM

Adam Roberts* ADAROB@GOOGLE.COM

Katherine Lee* KATHERINELEE@GOOGLE.COM

Sharan Narang SHARANNARANG@GOOGLE.COM

Michael Matena MMATENA@GOOGLE.COM

Yanqi Zhou YANQIZ@GOOGLE.COM

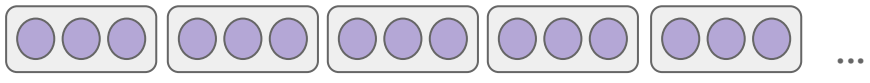
Wei Li MWEILI@GOOGLE.COM

Peter J. Liu PETERJLIU@GOOGLE.COM

Google, Mountain View, CA 94043, USA

T5 Pretraining: Span corruption

<X>, <Y>: sentinel tokens



Transformer encoder
(unmasked)

Thank you <X> me to your party <Y> week

<X> for inviting <Y> last <EOS>

Transformer decoder
(masked)

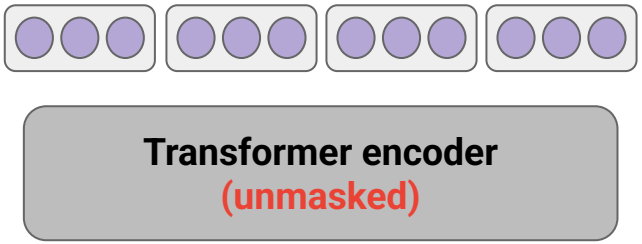
<BOS> <X> for inviting <Y> last

Thank you ~~for~~ ~~inviting~~ me to your party ~~last~~ week

encoder

decoder

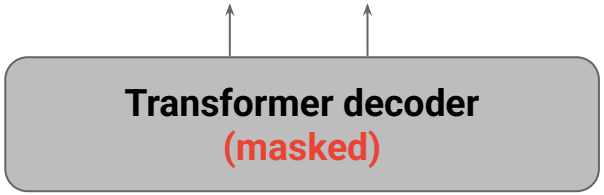
T5 Fine-tuning



sentiment analysis: this movie was good

encoder

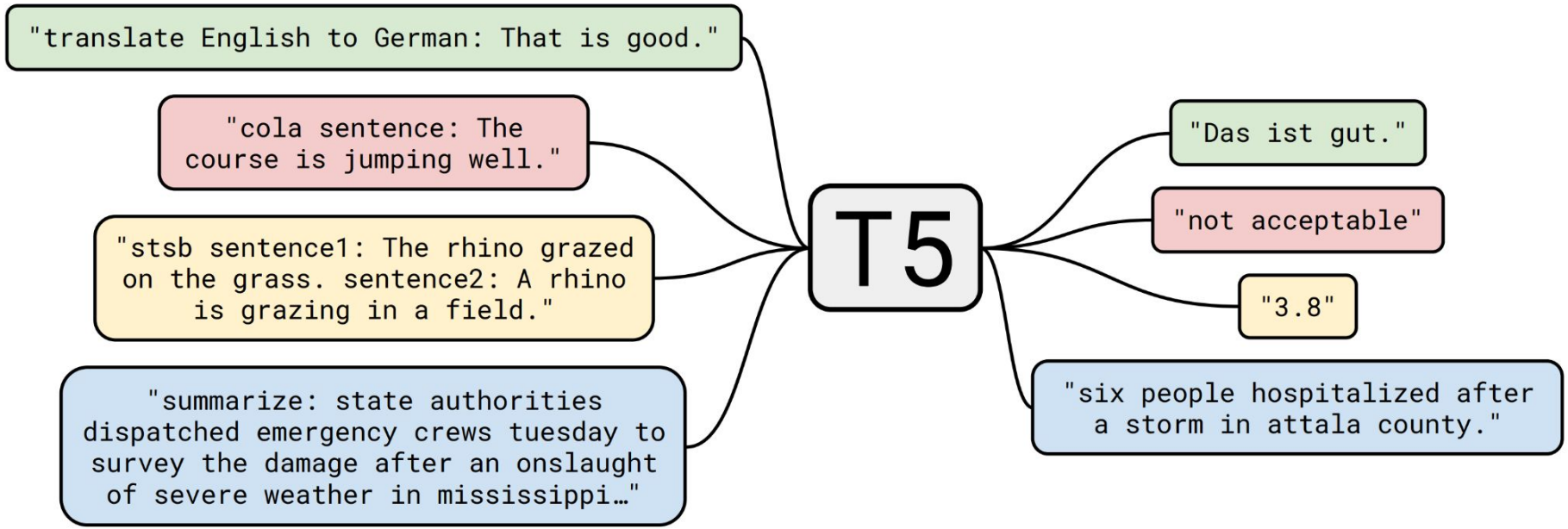
positive <EOS>



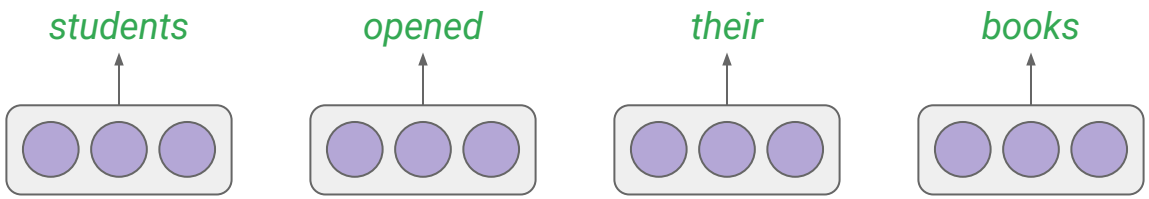
<BOS> positive

decoder

T5 facilitates multitask learning

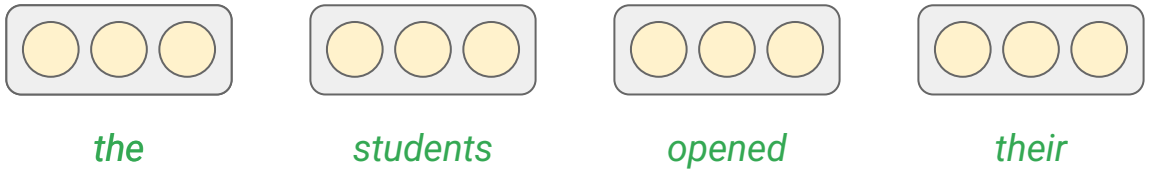


Decoder-only model



the architecture used in frontier LLMs

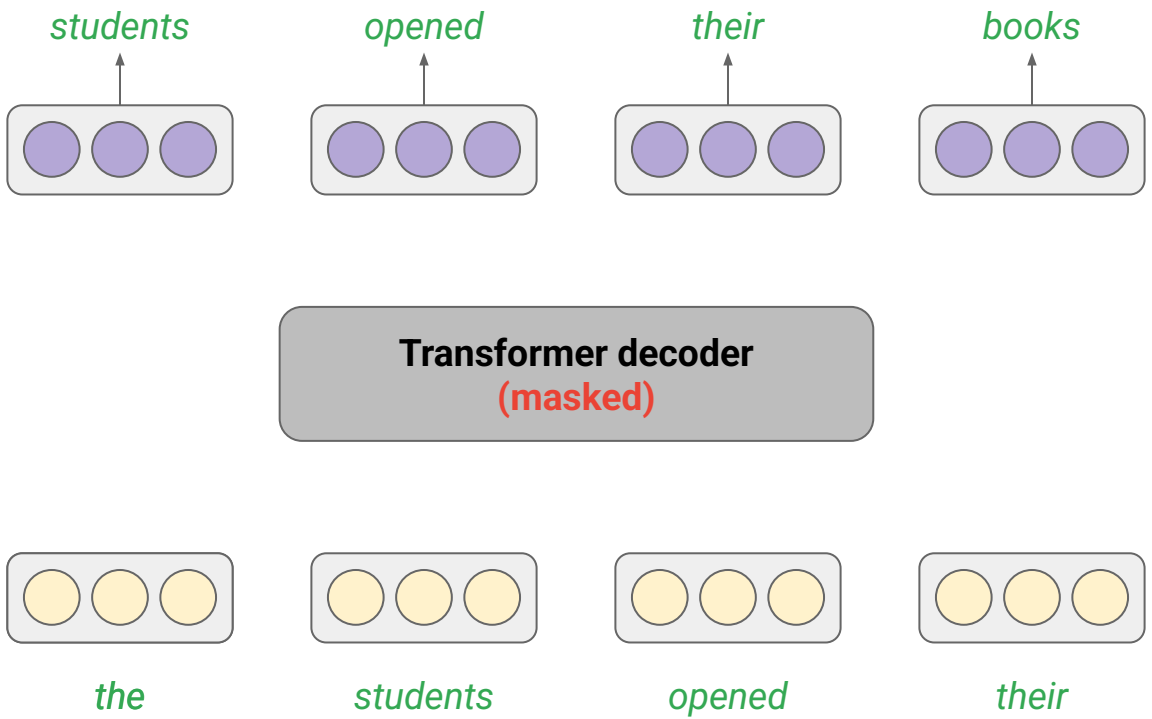
Transformer decoder
(masked)



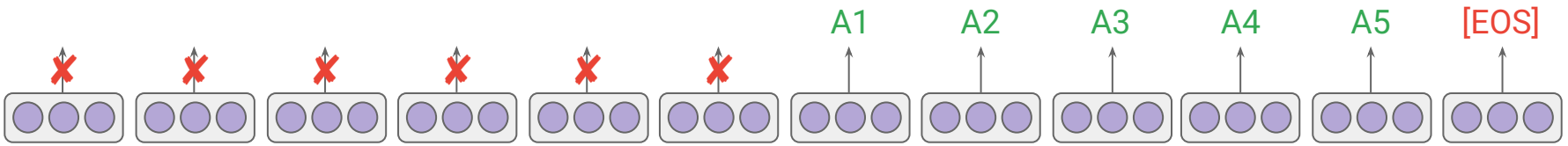
Note on cross-attention

- Can be used to inject non-text data (e.g., images, structured data, or even sensor readings) into the model

Pretraining with a causal LM (decoder-only)



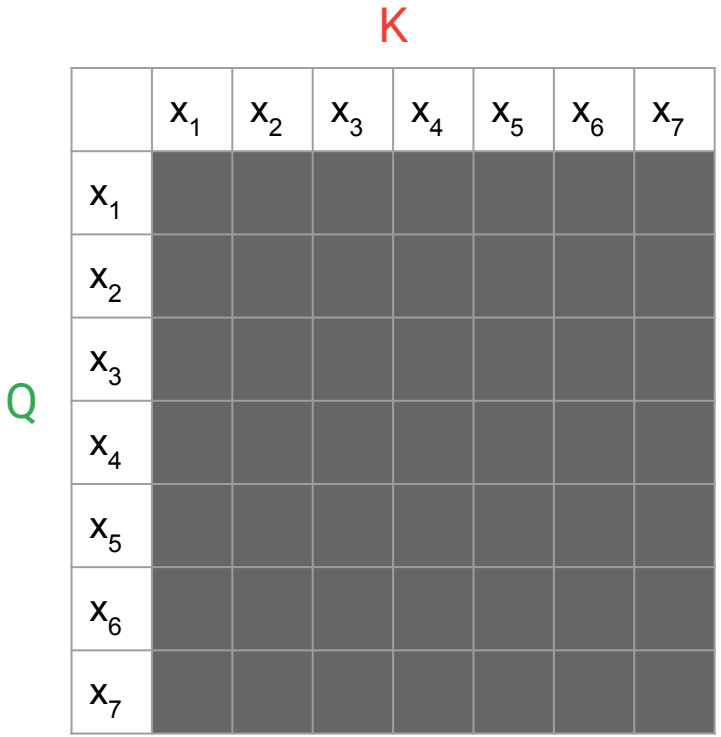
Training with prefix LM (decoder-only)



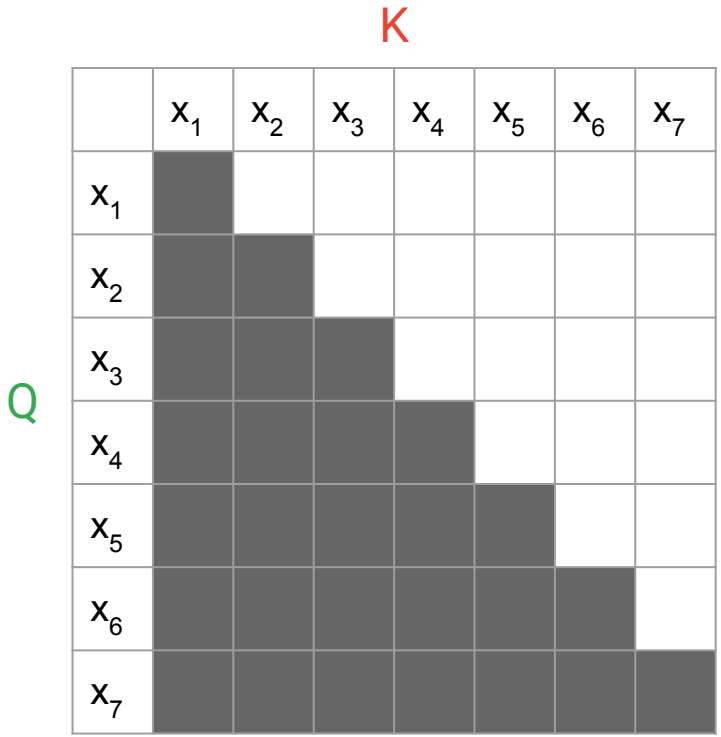
the architecture used in frontier LLMs

Transformer decoder
(partially masked)

Different attention mask patterns

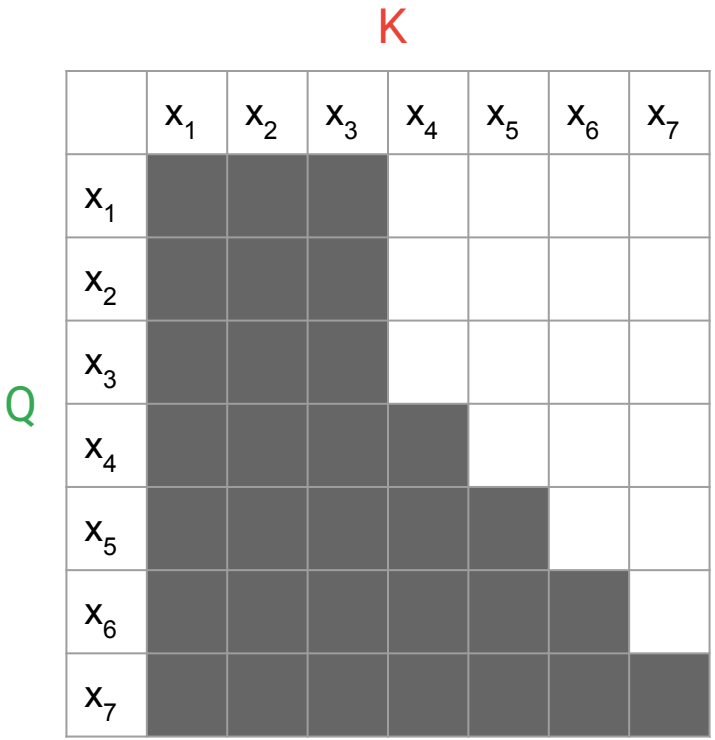


fully-visible



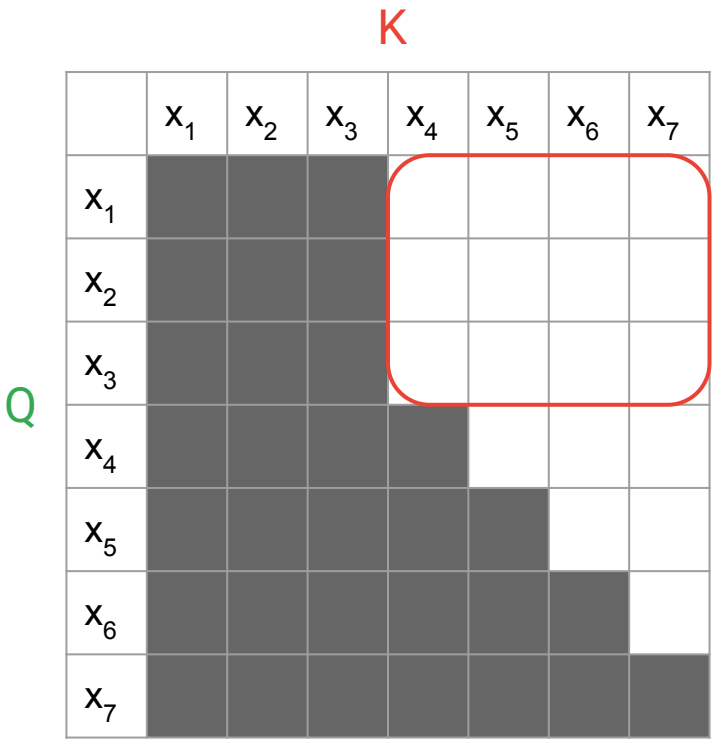
causal

Different attention mask patterns (cont'd)



Prefix LM

Different attention mask patterns (cont'd)



Why masking here?

Thank you!