Course introduction & Transformers

CS 6804: Frontier AI Systems

Spring 2026

https://tuvllms.github.io/ai-seminar-spring-2026/

Tu Vu

VIRGINIA TECH

Schedule and location

- **Time:** Monday & Wednesday 2:30 - 3:45 PM
- **Location:** Derring Hall 3092

Except for guest lectures, all lectures, student presentations, and discussions will be held in person only. No recordings will be made available.

Staff

- **Instructor:** Tu Vu
  - **Office hours:**
    - Friday 2:45 - 3:45 PM, D&DS 374
    - both in-person and via Zoom (link will be posted on Piazza)
    - will start next Friday, January 30th

- **Contact:** Please email *me* at cs6804instructors@gmail.com. For anonymous questions or comments, please use this form.

Course materials

- Slides and readings (usually published research papers) will be provided as PDFs on the course website https://tuvllms.github.io/ai-seminar-spring-2026/.

You don't need to purchase any textbooks!

Course materials (cont'd)

- Other useful texts
  - Speech and Language Processing by Jurafsky and Martin
  - Reinforcement Learning from Human Feedback by Lambert
  - Foundations of Large Language Models by Xiao and Zhu
  - Dive into Deep Learning by Zhang, Lipton, Li, and Smola
  - Deep Learning by Goodfellow, Bengio, and Courville

Communication channels

- **Course website:** https://tuvllms.github.io/ai-seminar-spring-2026/
- **Piazza**: announcements and discussions
- **Gradescope**: assignment submissions
- **Canvas**: final grades & others
- **Discord**: questions & discussions (invite link will be available on Piazza)

Prerequisites

- No prerequisites are required for this course; however, the following could be helpful:

  - Familiarity with basic machine learning concepts
  - Familiarity with basic statistical concepts
  - Proficiency in Python programming

Grading policy

- **Grading breakdown:**
  - Written homework assignments (20%)
  - Discussion question submission + in-class participation (25%)
  - Presentations of assigned papers (25%)
  - Exam (in-class, 30%)
- Student presentations: groups of 2-3; all groups should be formed by January 30th

- Each student is allowed ***two*** late days total across all homework submissions

AI assistance policy

- AI assistance is permitted for completing assignments.
- If you use AI tools like ChatGPT or Gemini, you must submit the prompts you used and describe how the AI contributed to your work.
- It is your responsibility to verify the AI-generated content for accuracy before submission.

Course enrollment

- Please contact Sara Coulson at sara83@vt.edu with such requests
- The force-add request window for graduate-level courses https://students.cs.vt.edu/Graduate/forceadd.html
- This class is currently full though

This course

- Six weeks of me lecturing, so that we are all roughly on the same page
- Rest of semester: student presentations and discussions of assigned papers

Main topics

- Transformers & Pretraining Scaling
- Efficient training & inference
- Post-training & Reinforcement Learning
- Large reasoning models & Test-time scaling
- Agents & Compound AI systems

Focus:

- **Efficiency** & **Reasoning** & **Agentic AI**

Why do we study AI?

- one of the most interesting and fastest-growing field
- AI expert Kai-Fu Lee predicts that its impact will be “more than anything in the history of mankind”
- Moreover, the intellectual frontiers of AI are wide open. Whereas a student of an older science such as physics might feel that the best ideas have already been discovered by Galileo, Newton, Curie, Einstein, and the rest, AI still has many openings for full-time masterminds

Russell and Norvig, Chapter 1

AI is creating new billionaires at a record pace

AI Boom Billionaires: These Tech Moguls Joined The Forbes List In 2025

The New York Times
https://www.nytimes.com › ai-researchers-nba-stars

A.I. Researchers Are Negotiating $250 Million Pay ...

Aug 1, 2025 — A.I. Researchers Are Negotiating $250 Million Pay Packages. Just Like NBA Stars. - The New York Times.

Heroes of AI / Deep Learning

Geoffrey Hinton

Backpropagation & deep learning

Yann LeCun

Convolutional neural networks

Yoshua Bengio

Representation learning

Juergen Schmidhuber

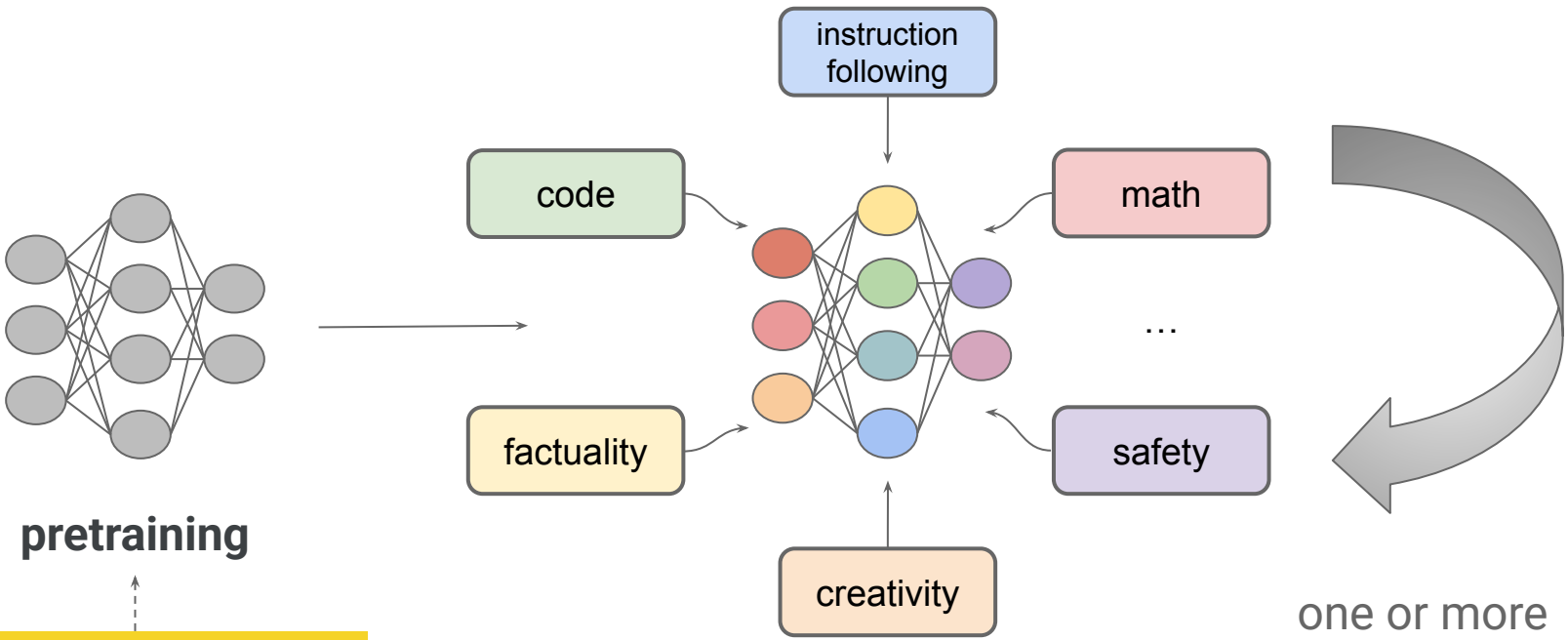
Long short-term memory (LSTM)

Andrew Ng

AI education & large-scale ML systems

Juergen Schmidhuber's TED talk in 2012

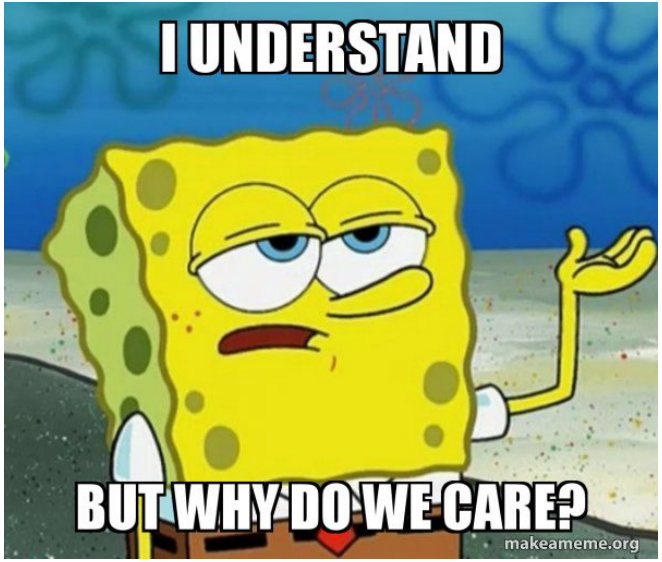
The development of modern LLMs



Language modeling

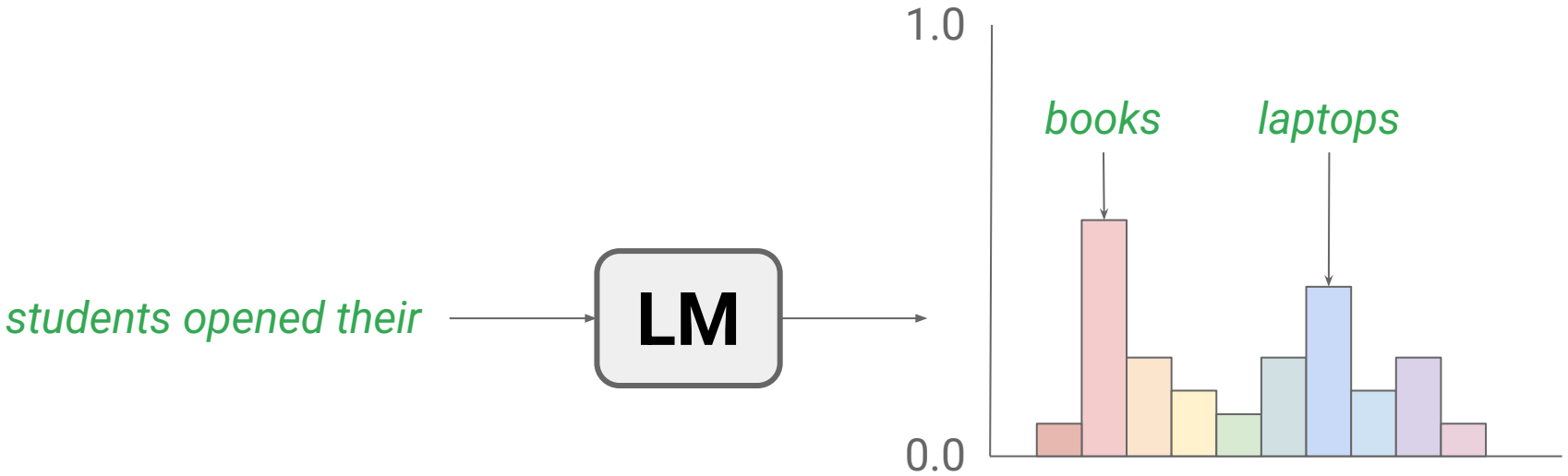
- **Predicting the next/missing word**

  Example: “The cat is on the ___.” → Predicted: “mat”.



What is a language model?

- A machine learning model that assigns a ***probability*** to each possible next word, or a ***probability distribution*** over possible next words



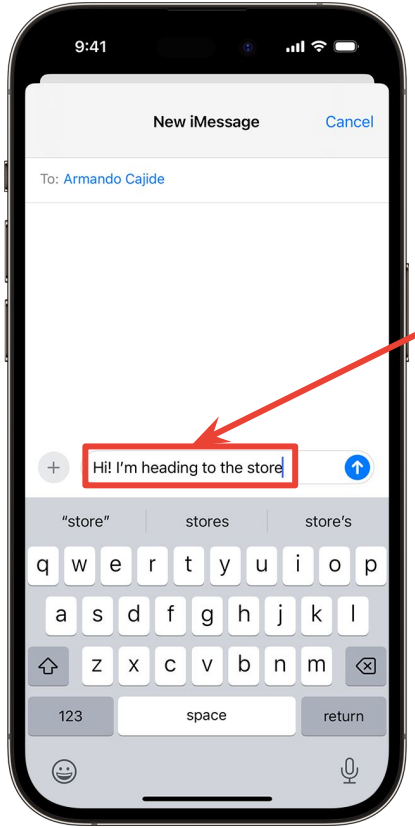
What is a language model? (cont'd)

- A language model can also assign a probability to an entire sentence

$$P(w_1, w_2, \ldots, w_n) = P(w_1) \times P(w_2|w_1) \times \ldots \times P(w_n|w_1, w_2, \ldots, w_{n-1})$$

P(*"The cat is on the mat"*) = ***P***(*"The"*) *x* ***P***(*"cat" | "The"*) *x* ***P***(*"is" | "The cat"*) *x* ***P***(*"on" | "The cat is"*) *x* ***P***(*"the" | "The cat is on"*) *x* ***P***(*"mat" | "The cat is on the"*)

You use language models everyday!



prefix

Google

san f

san francisco weather
san francisco
san francisco giants
san fernando valley
san francisco state university
san francisco hotels
san francisco 49ers
san fernando
san fernando mission
san francisco zip code

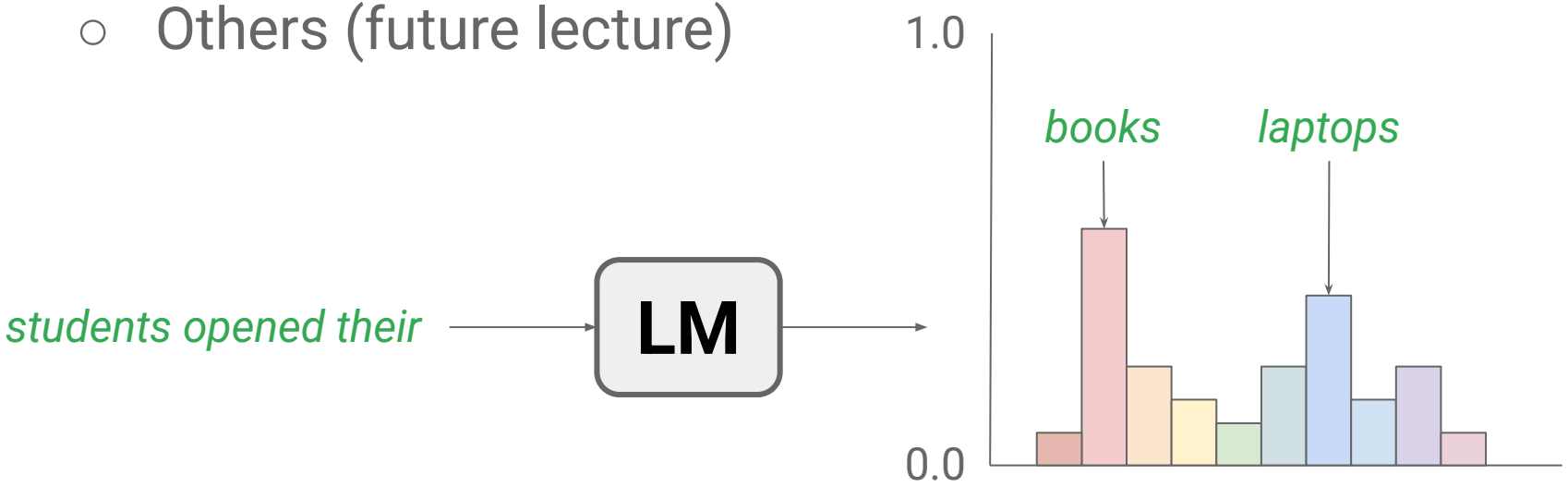
Google Search | I'm Feeling Lucky

source: Apple Support

source: Google Blog

How to sample sentences from a language model?

- Decoding strategies
  - Greedy decoding
  - Sampling
  - Others (future lecture)



N-grams

- An n-gram is a sequence of n words
- Unigram (n=1)
  - “The”, “water”, “of”, “Walden”, “Pond”
- Bigram (n=2)
  - “The water”, “water of”, “of Walden”, “Pond”
- Trigram (n=3)
  - “The water of”, “water of Walden”, “of Walden Pond”
- 4-gram
- …

Matrix-vector multiplication

Matrix A (dimensions 4×3):

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

Vector x (dimensions 3×1):

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Resulting vector b (dimensions 4×1):

$$b = A \cdot x = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 \\ a_{41}x_1 + a_{42}x_2 + a_{43}x_3 \end{bmatrix}$$

Softmax function

For a vector $y = [y_1, y_2, \ldots, y_V]$ of dimension V, the softmax transformation is calculated as:

$$\text{softmax}(y) = \left[\frac{e^{y_1}}{\sum e^y}, \frac{e^{y_2}}{\sum e^y}, \ldots, \frac{e^{y_V}}{\sum e^y} \right]$$

where $\sum e^y = e^{y_1} + e^{y_2} + \cdots + e^{y_V}$.

Word representations / embeddings

- High-dimensional / sparse / one-hot representations
- Low-dimensional / dense representations

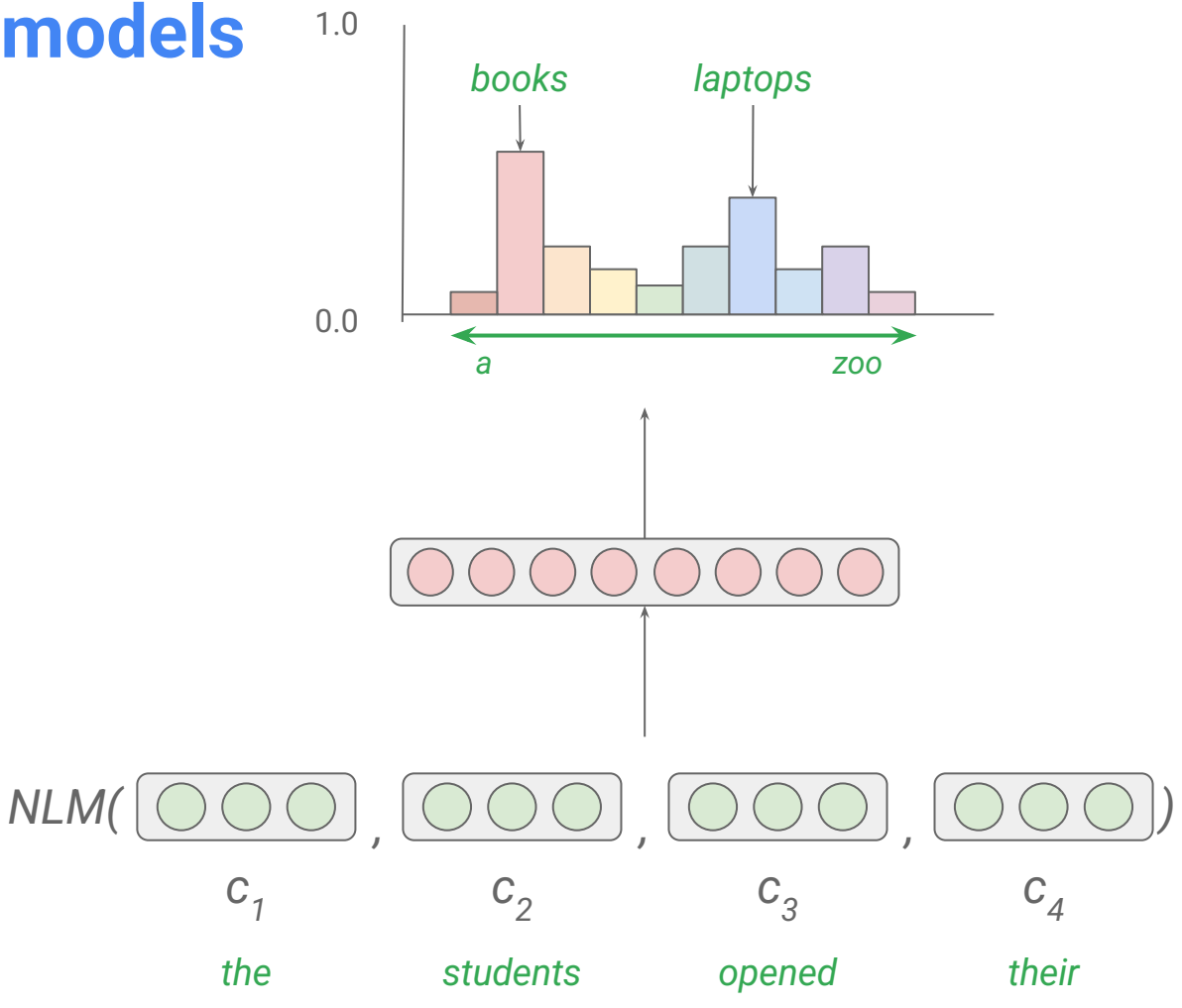
Word representations / embeddings (cont'd)

```
#What is the vector representation for a word?
w2v_model['computer']
```

```
array([ 1.07421875e-01, -2.01171875e-01,  1.23046875e-01,  2.11914062e-01,
       -9.13085938e-02,  2.16796875e-01, -1.31835938e-01,  8.30078125e-02,
        2.02148438e-01,  4.78515625e-02,  3.66210938e-02, -2.45361328e-02,
        2.39257812e-02, -1.60156250e-01, -2.61230469e-02,  9.71679688e-02,
       -6.34765625e-02,  1.84570312e-01,  1.70898438e-01, -1.63085938e-01,
       -1.09375000e-01,  1.49414062e-01, -4.65393066e-04,  9.61914062e-02,
        1.68945312e-01,  2.60925293e-03,  8.93554688e-02,  6.49414062e-02,
        3.56445312e-02, -6.93359375e-02, -1.46484375e-01, -1.21093750e-01,
       -2.27539062e-01,  2.45361328e-02, -1.24511719e-01, -3.18359375e-01,
       -2.20703125e-01,  1.30859375e-01,  3.66210938e-02, -3.63769531e-02,
       -1.13281250e-01,  1.95312500e-01,  9.76562500e-02,  1.26953125e-01,
        6.59179688e-02,  6.93359375e-02,  1.02539062e-02,  1.75781250e-01,
       -1.68945312e-01,  1.21307373e-03, -2.98828125e-01, -1.15234375e-01,
        5.66406250e-02, -1.77734375e-01, -2.08984375e-01,  1.76757812e-01,
        2.38037109e-02, -2.57812500e-01, -4.46777344e-02,  1.88476562e-01,
        5.51757812e-02,  5.02929688e-02, -1.06933594e-01,  1.89453125e-01,
       -1.16210938e-01,  8.49609375e-02, -1.71875000e-01,  2.45117188e-01,
       -1.73828125e-01, -8.30078125e-03,  4.56542969e-02, -1.61132812e-02,
        1.86523438e-01, -6.05468750e-02, -4.17480469e-02,  1.82617188e-01,
        2.20703125e-01, -1.22558594e-01, -2.55126953e-02, -3.08593750e-01,
        9.13085938e-02,  1.60156250e-01,  1.70898438e-01,  1.19628906e-01,
        7.08007812e-02, -2.64892578e-02, -3.08837891e-02,  4.06250000e-01,
       -1.01562500e-01,  5.71289062e-02, -7.26318359e-03, -9.17968750e-02,
       -1.50390625e-01, -2.55859375e-01,  2.16796875e-01, -3.63769531e-02,
        2.24609375e-01,  8.00781250e-02,  1.56250000e-01,  5.27343750e-02,
```

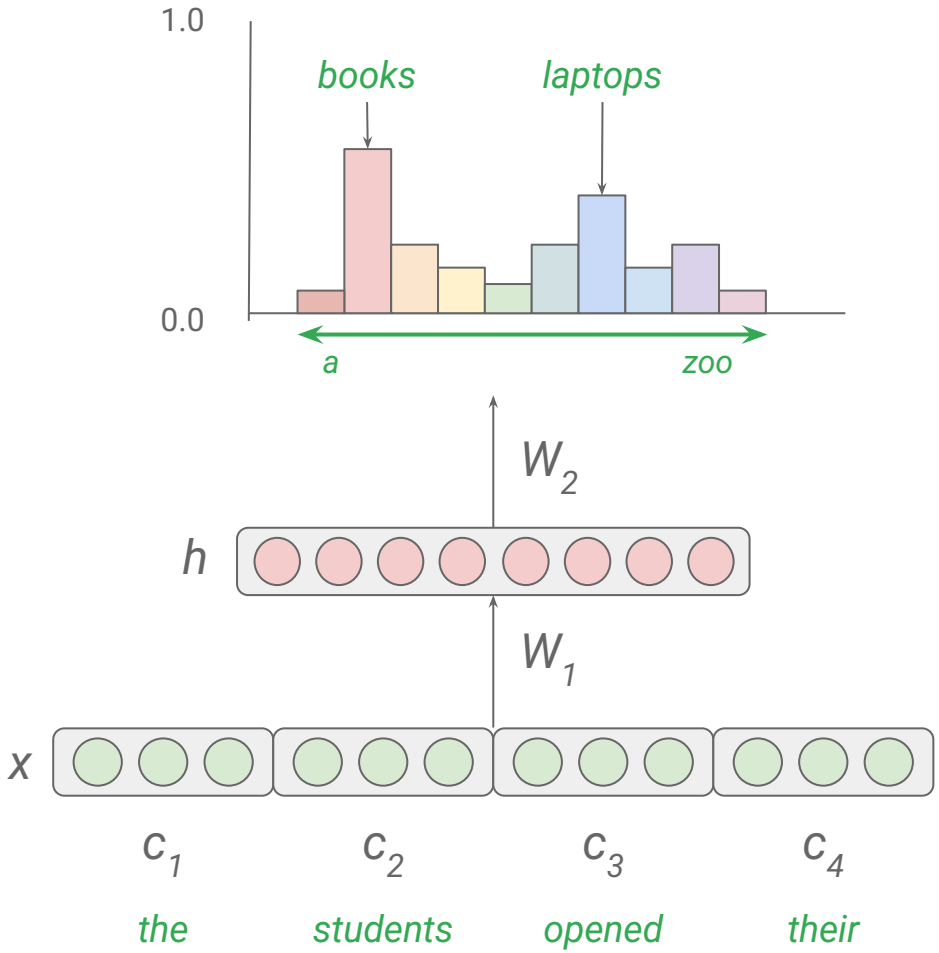
Connected to Python 3 Google Compute Engine backend

Neural language models



output distribution

$\hat{y} = softmax(W_2 h)$



Composition functions

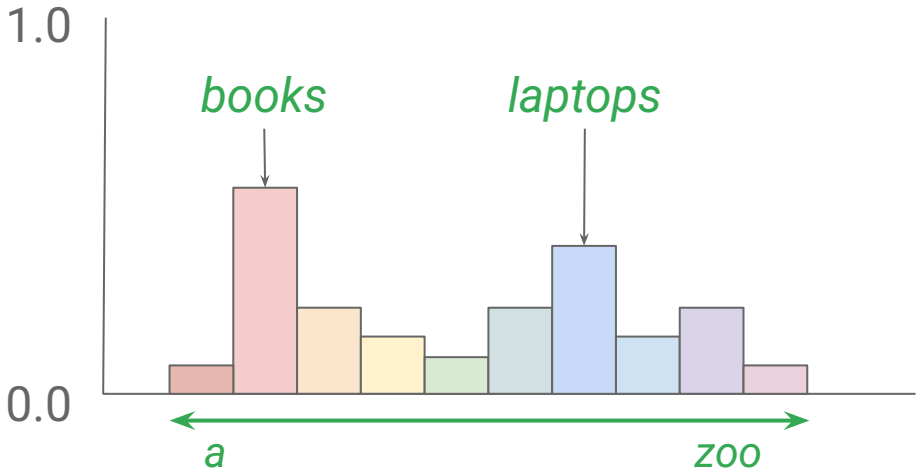
- Element-wise functions
  - e.g., just sum up all of the word embeddings
- Concatenation
- Feedforward neural networks
- Convolutional neural networks
- Recurrent neural networks
- Transformers

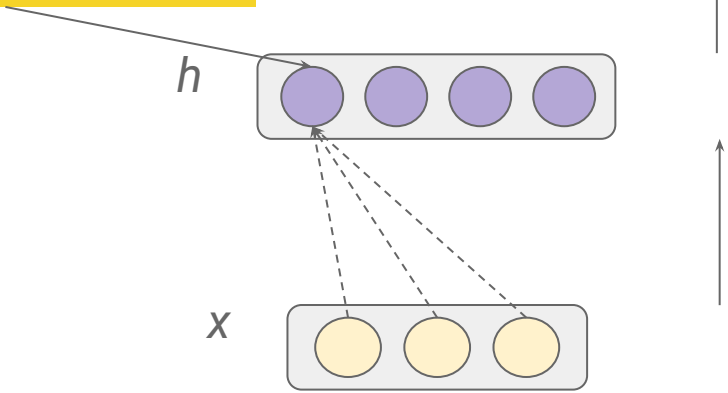
Feedforward neural language model

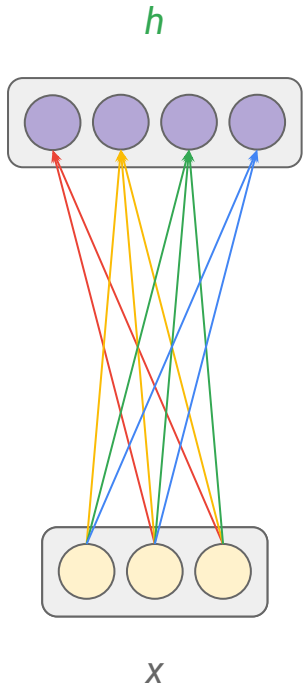
hidden layer

$h = f(W_1 x)$

hidden unit: taking a weighted sum of its inputs and then applying a non-linearity







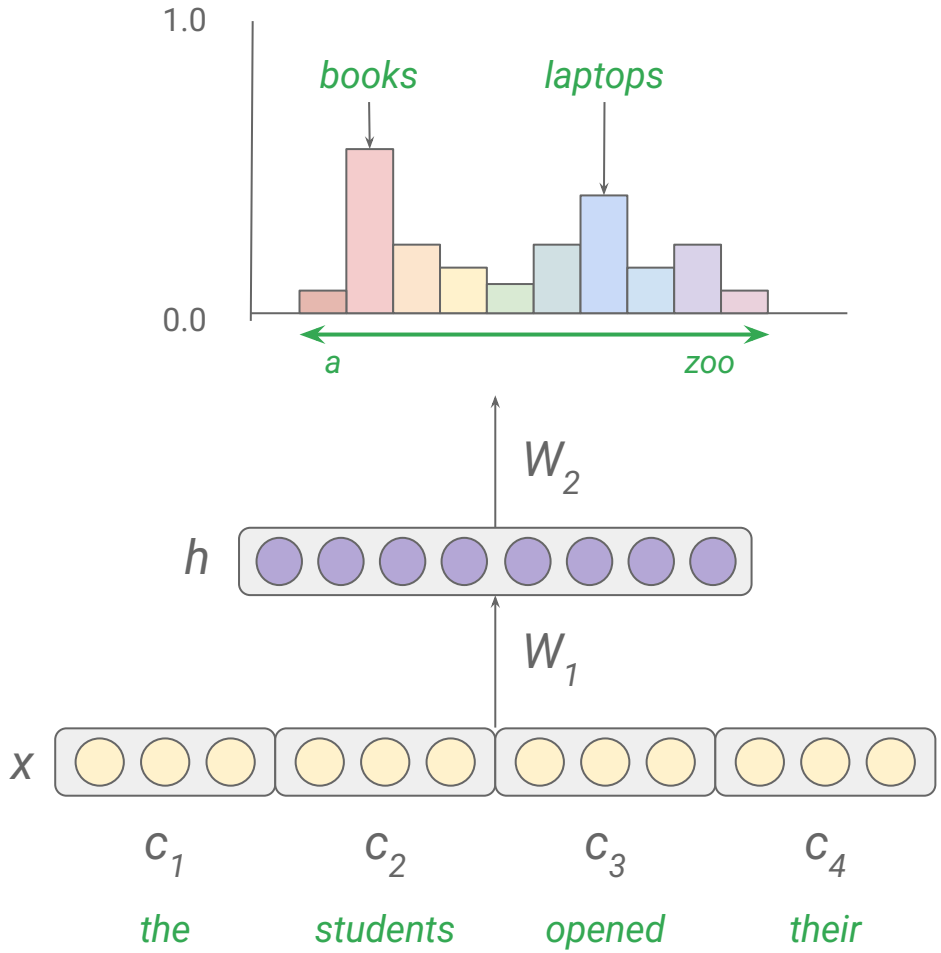
hidden layer

$h = f(W_1 x)$

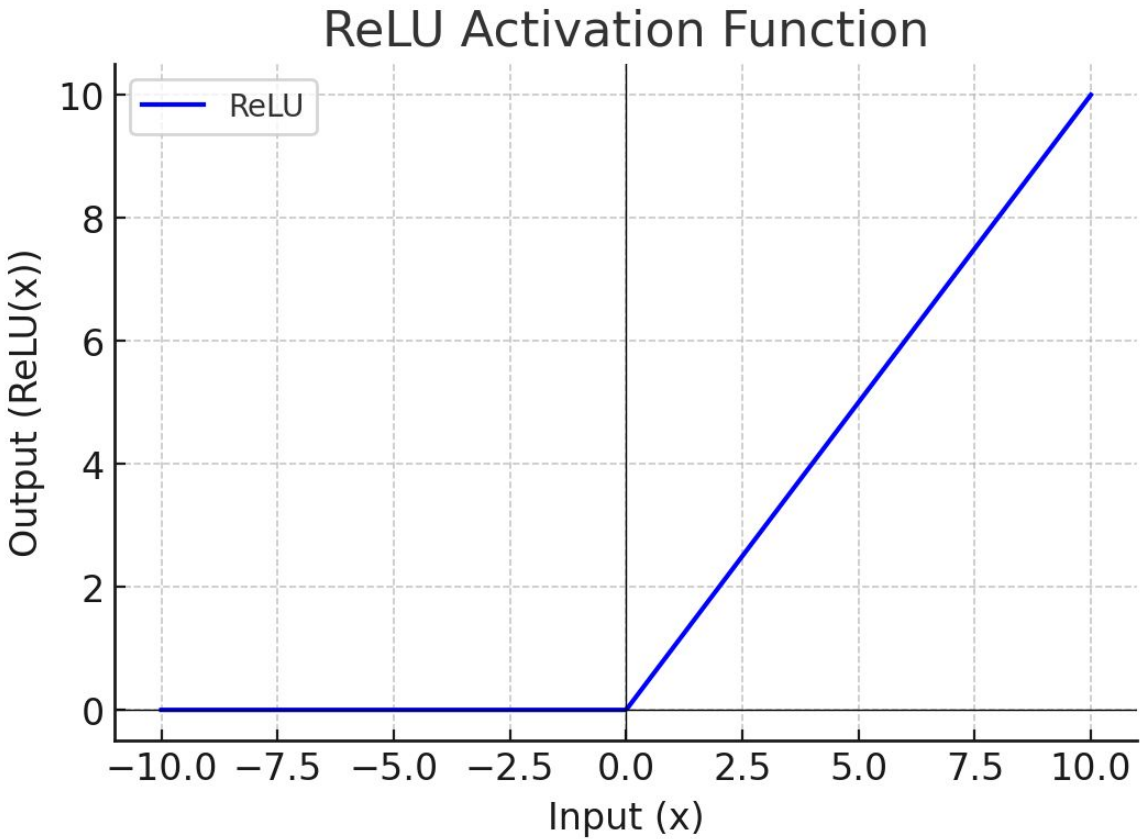
f is a non-linear activation function to model non-linear relationships between words

output distribution

$\hat{y} = softmax(W_2 h)$

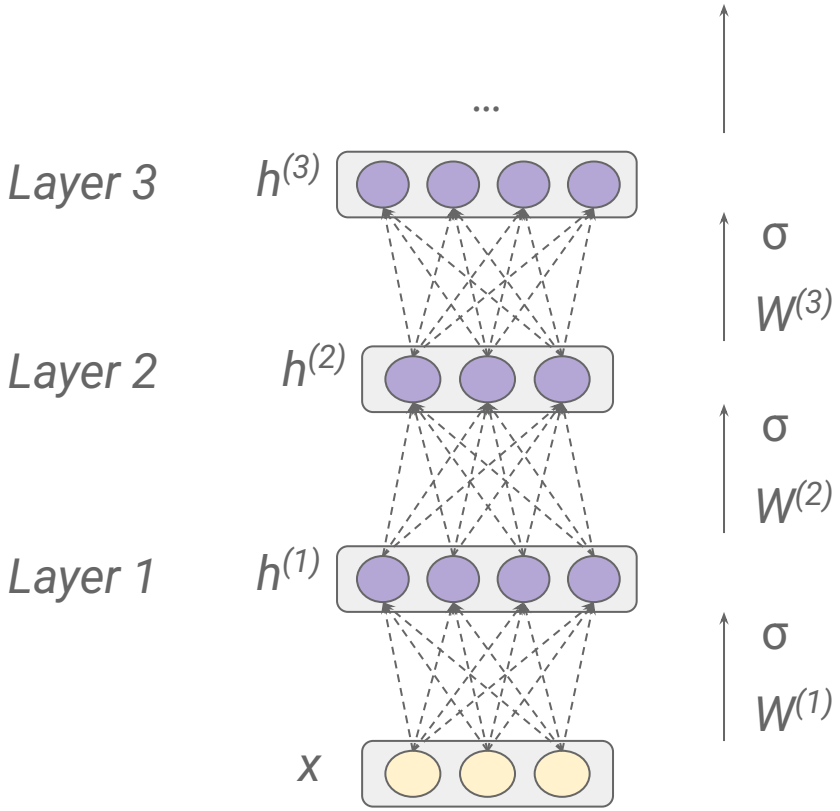


Activation functions



Rectified Linear Unit (ReLU)

Deep neural networks



hierarchical representations, where each layer builds upon the previous one

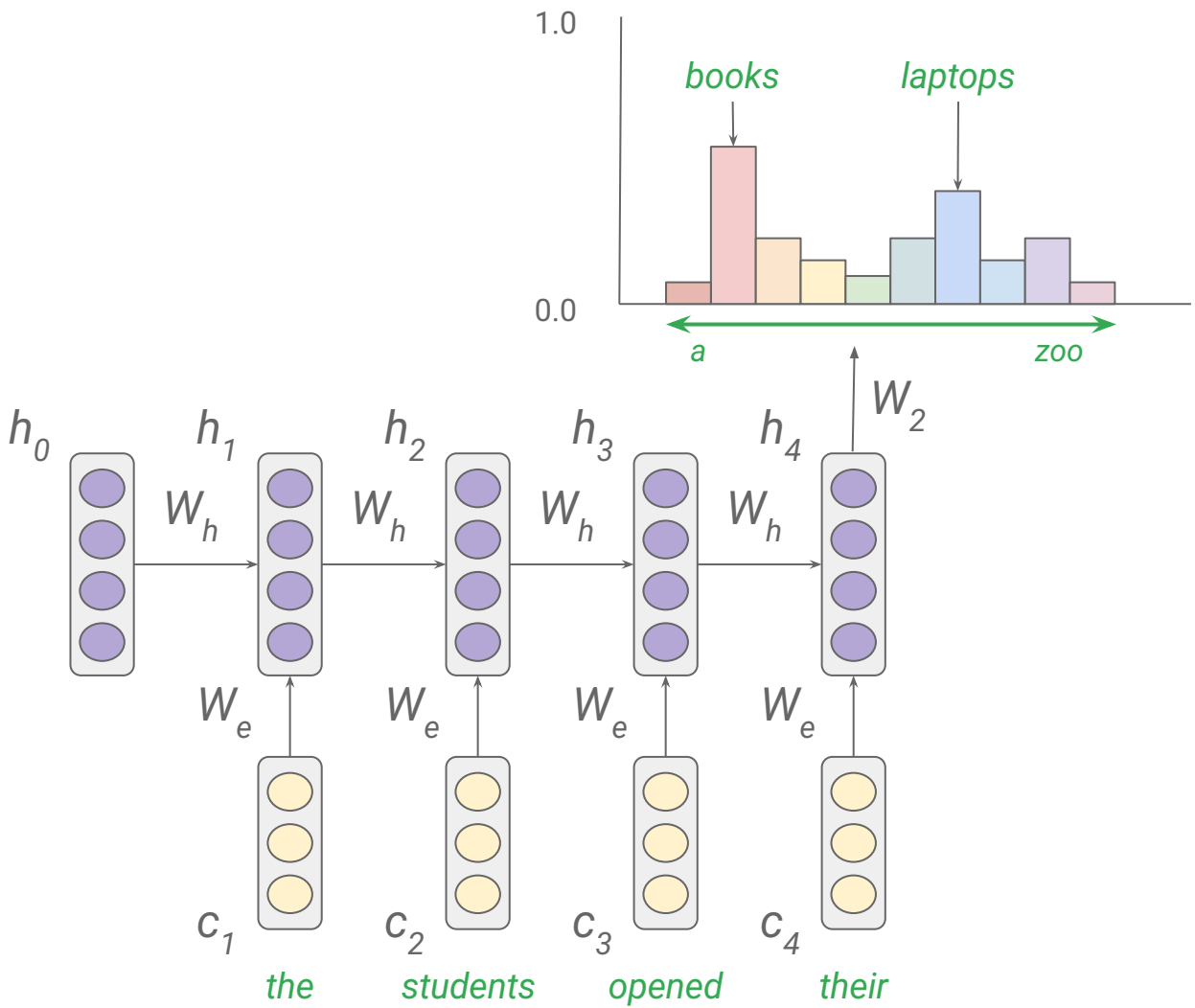
Recurrent neural networks (RNNs)

hidden states

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

output distribution

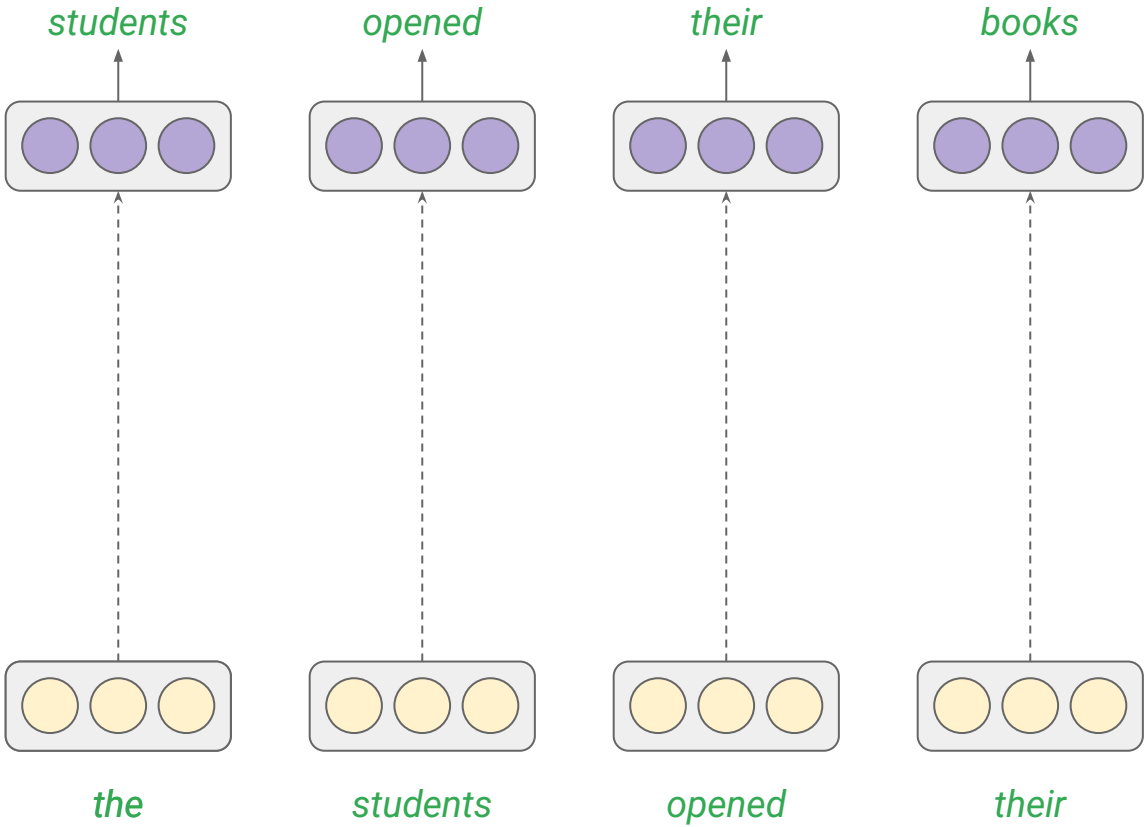
$$\hat{y} = softmax(W_2 h^{(n-1)})$$



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

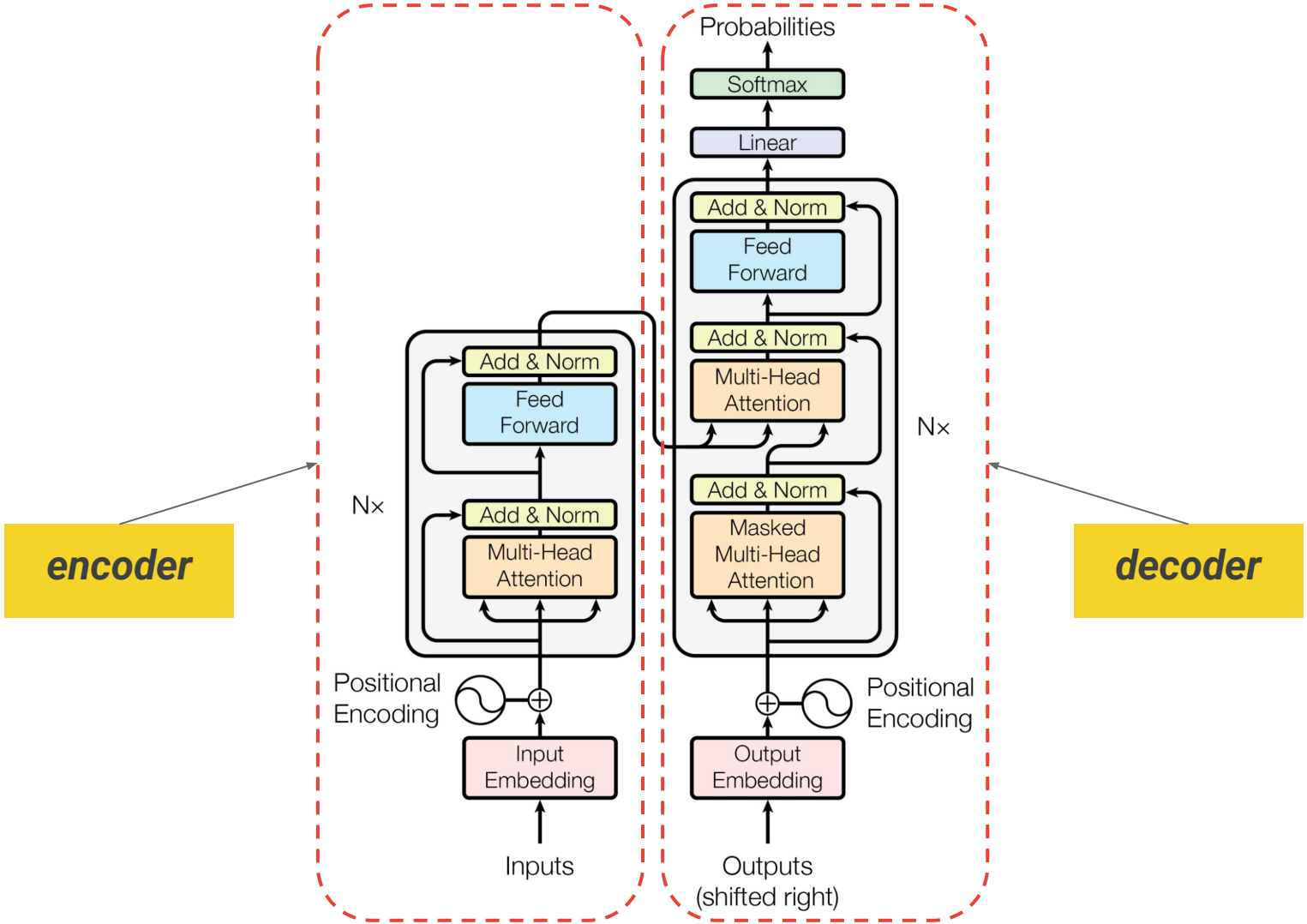
Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

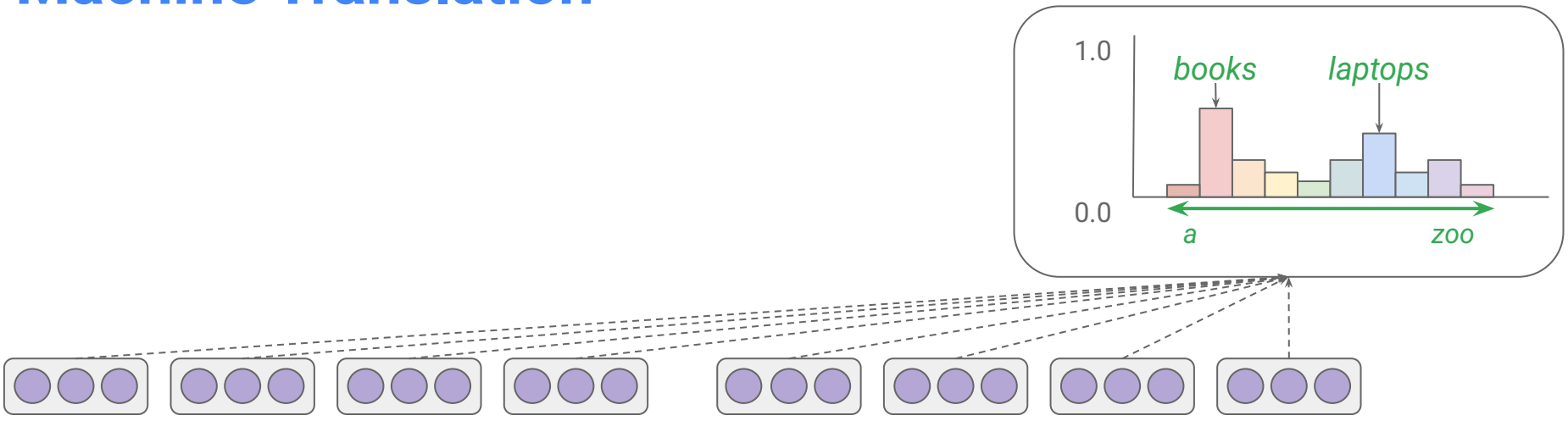
Transformers

- 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
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

Machine Translation



les étudiants ont ouvert leurs livres

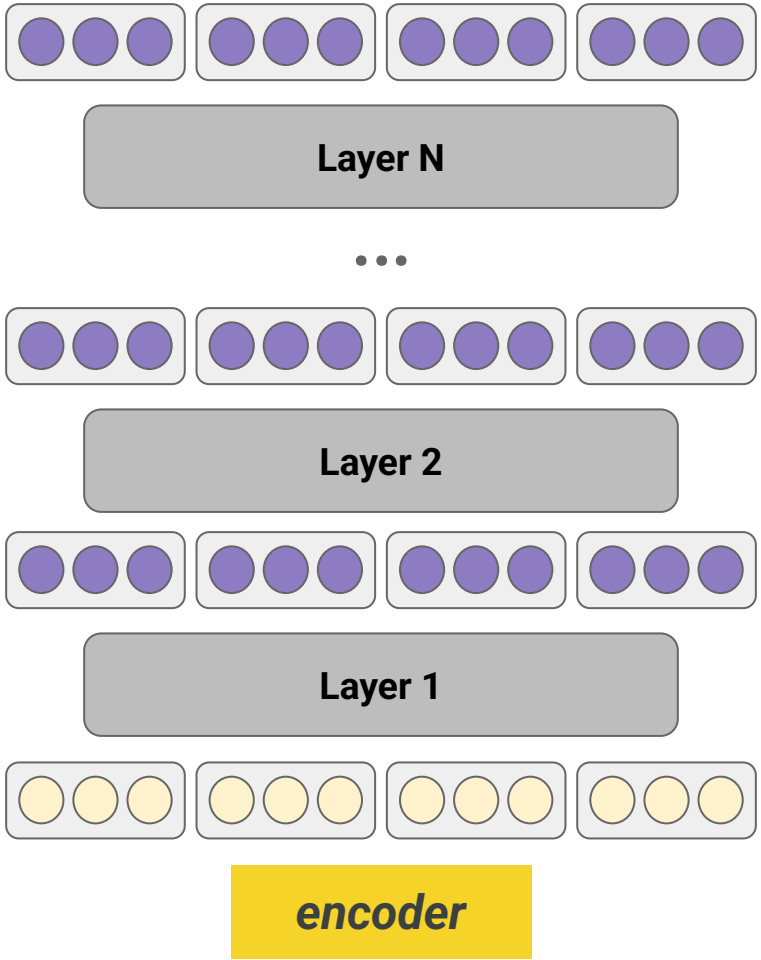
the *students* *opened* *their*

encoder ***decoder***

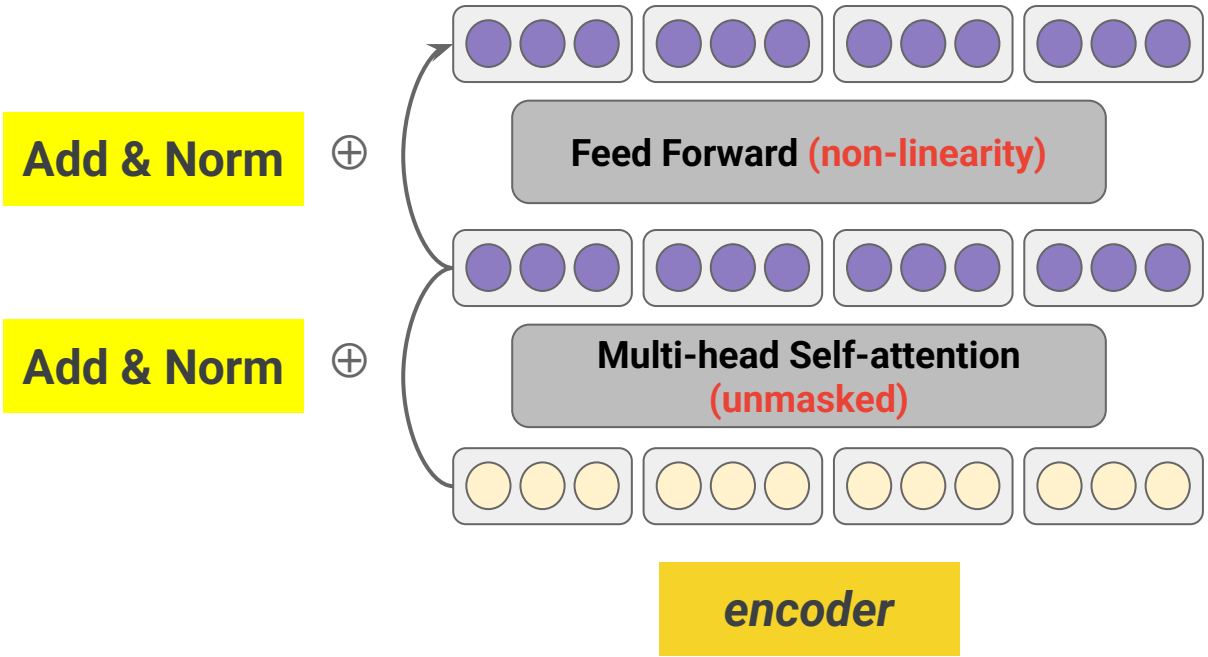
Different model architectures

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

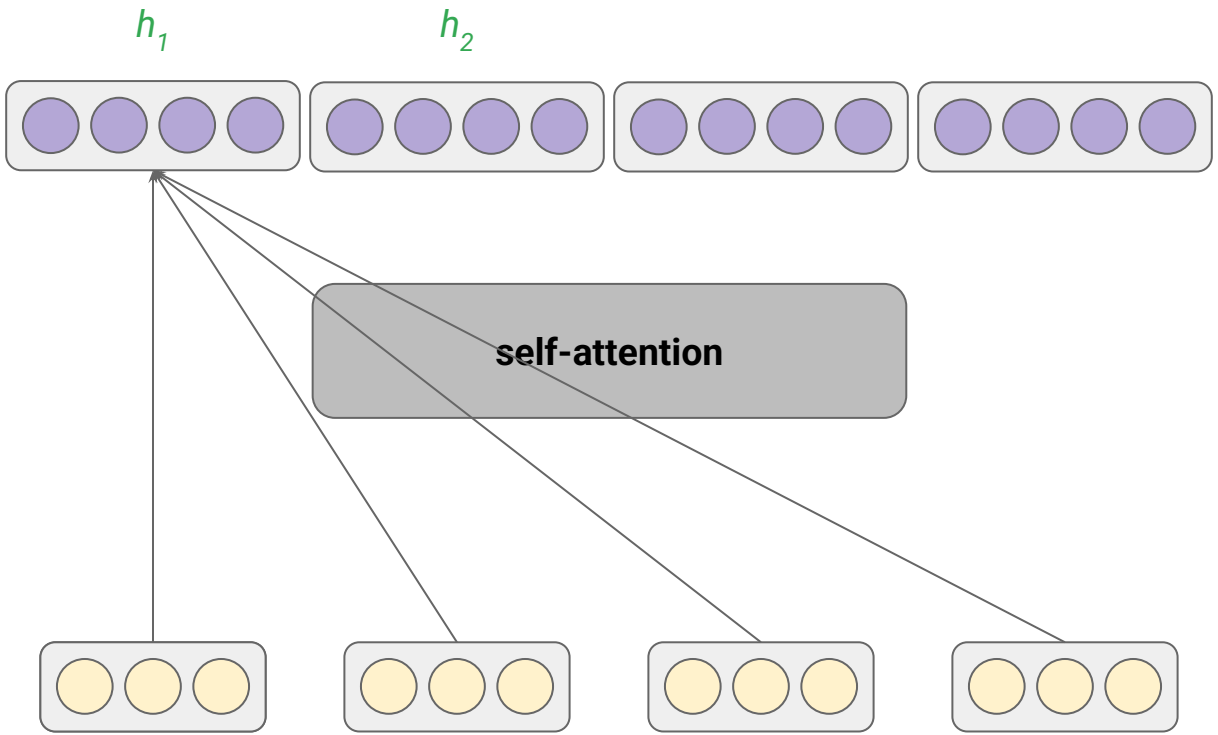
Encoder (N layers)



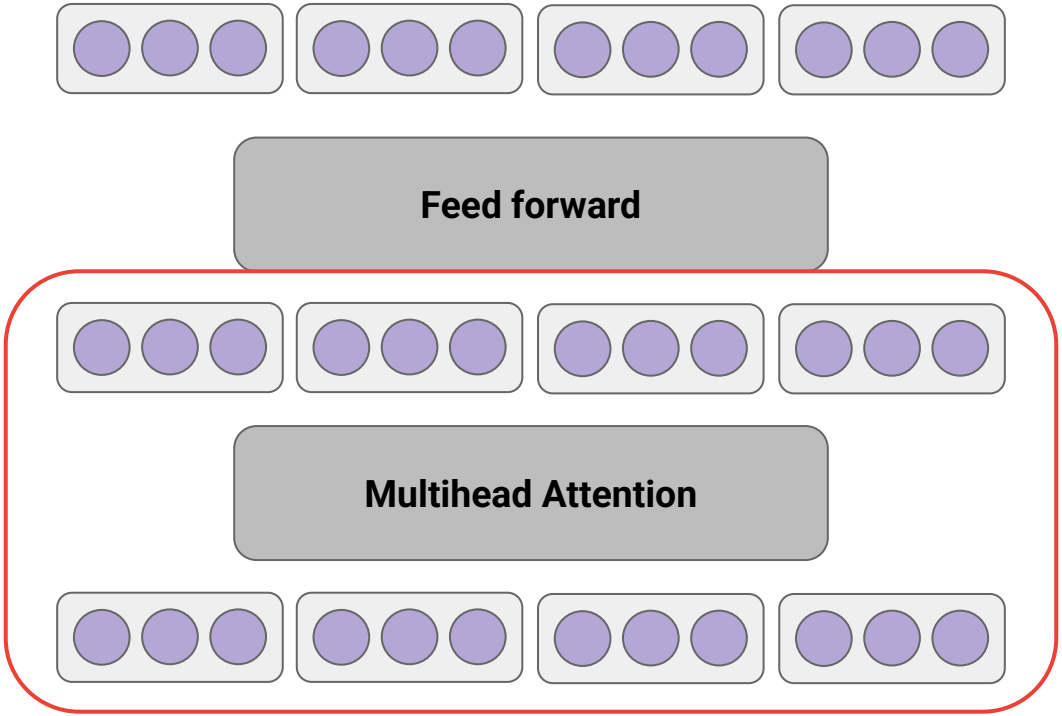
Encoder (one layer)



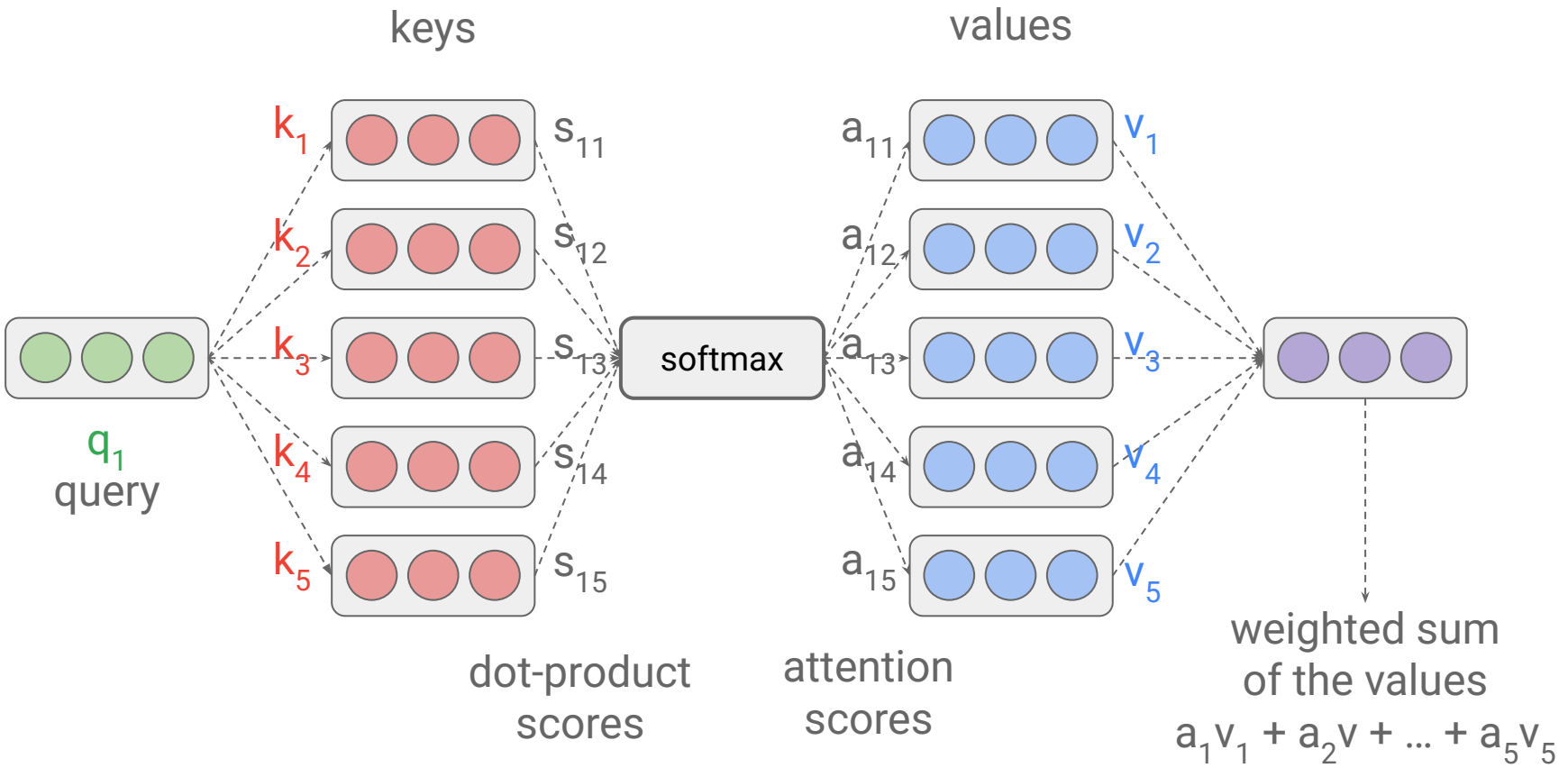
Self-attention



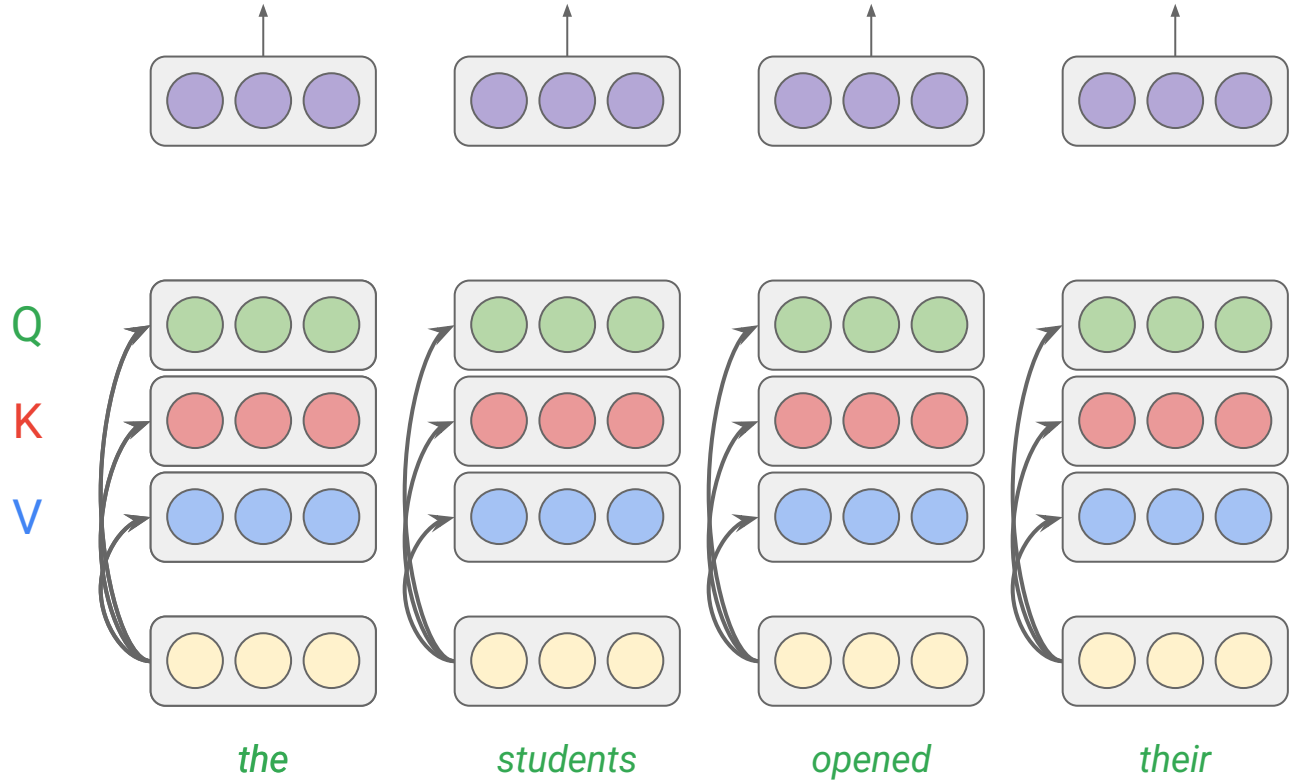
Transformer block



Attention


keys
values
k_1
k_2
k_3
k_4
k_5
s_{11}
s_{12}
s_{13}
s_{14}
s_{15}
softmax
a_{11}
a_{12}
a_{13}
a_{14}
a_{15}
v_1
v_2
v_3
v_4
v_5
q_1
query
dot-product scores
attention scores
weighted sum of the values
$a_1v_1 + a_2v + \ldots + a_5v_5$

Attention (cont'd)



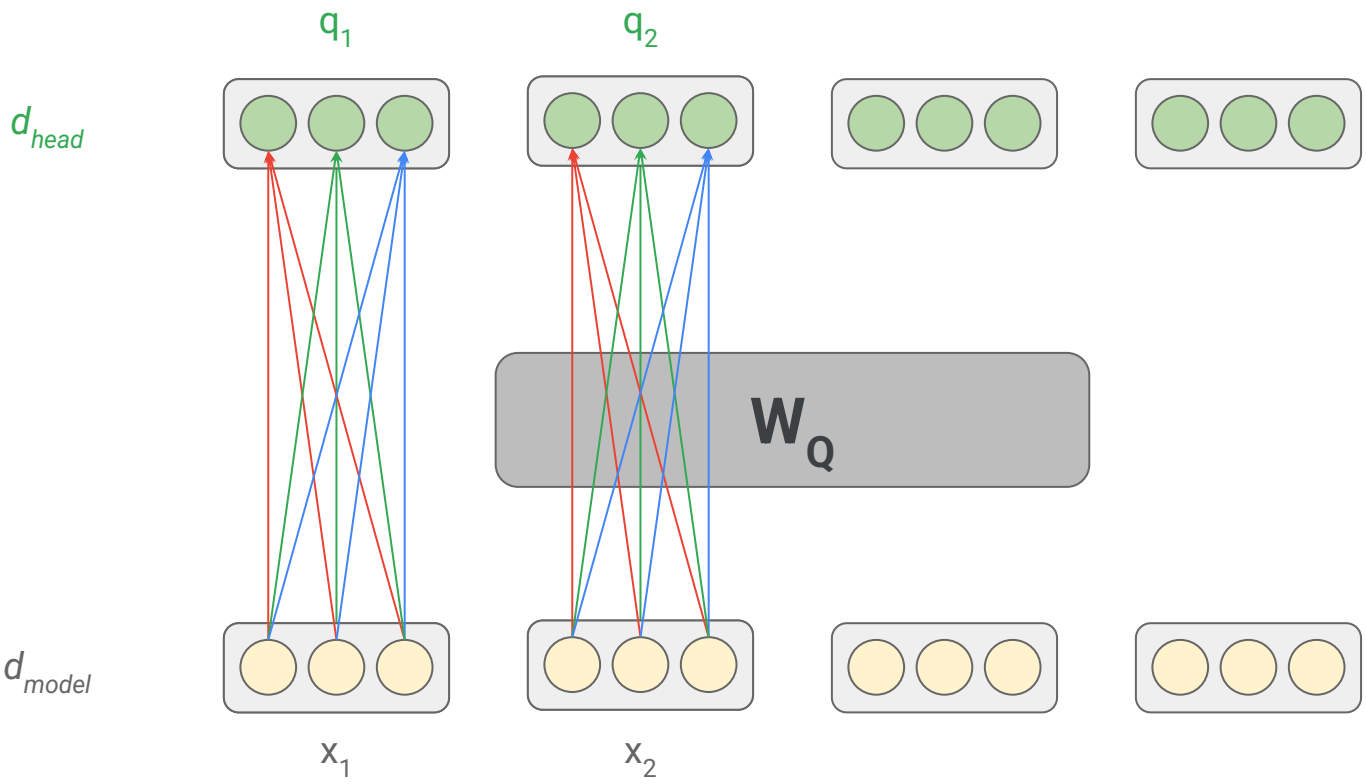
$$Q = X \cdot W_Q$$

$$K = X \cdot W_K$$

$$V = X \cdot W_V$$

linear projections

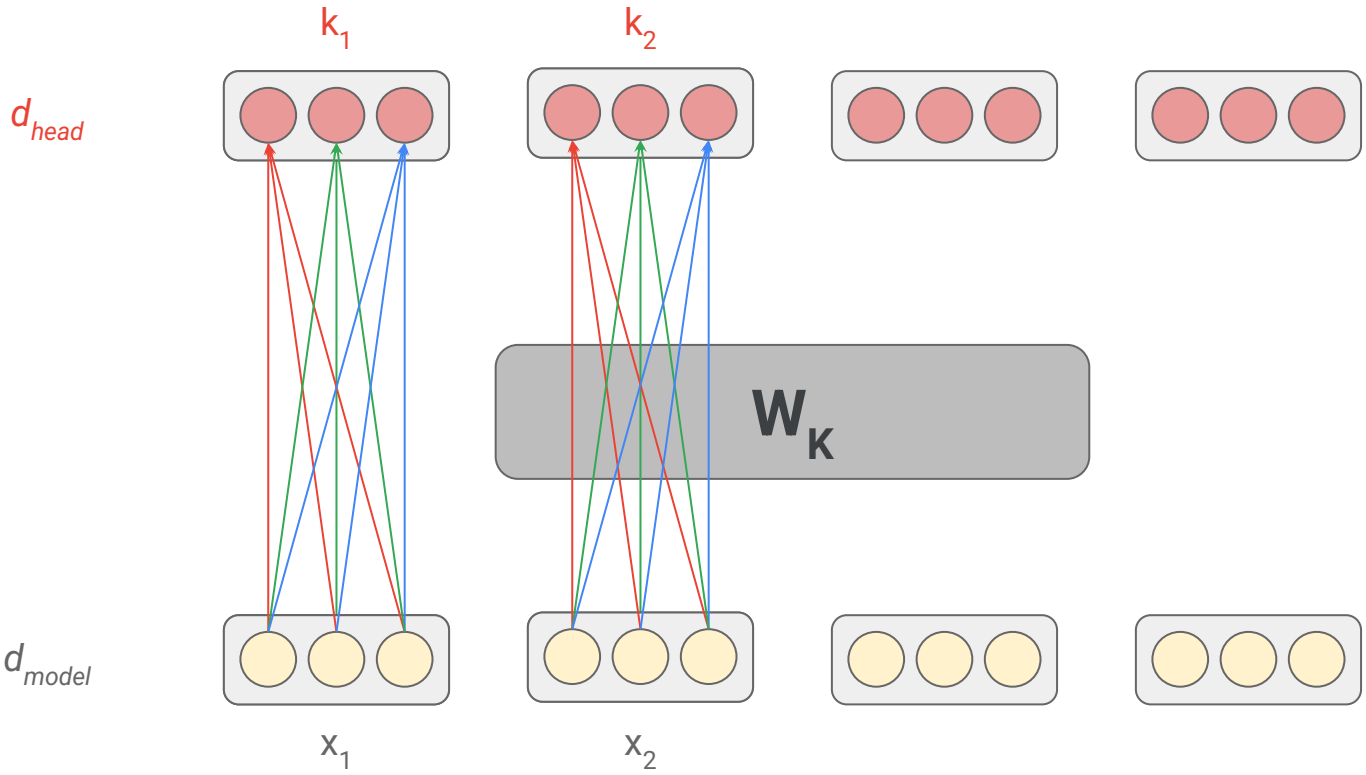
Query vectors



$$Q = X \cdot W_Q$$

linear projections

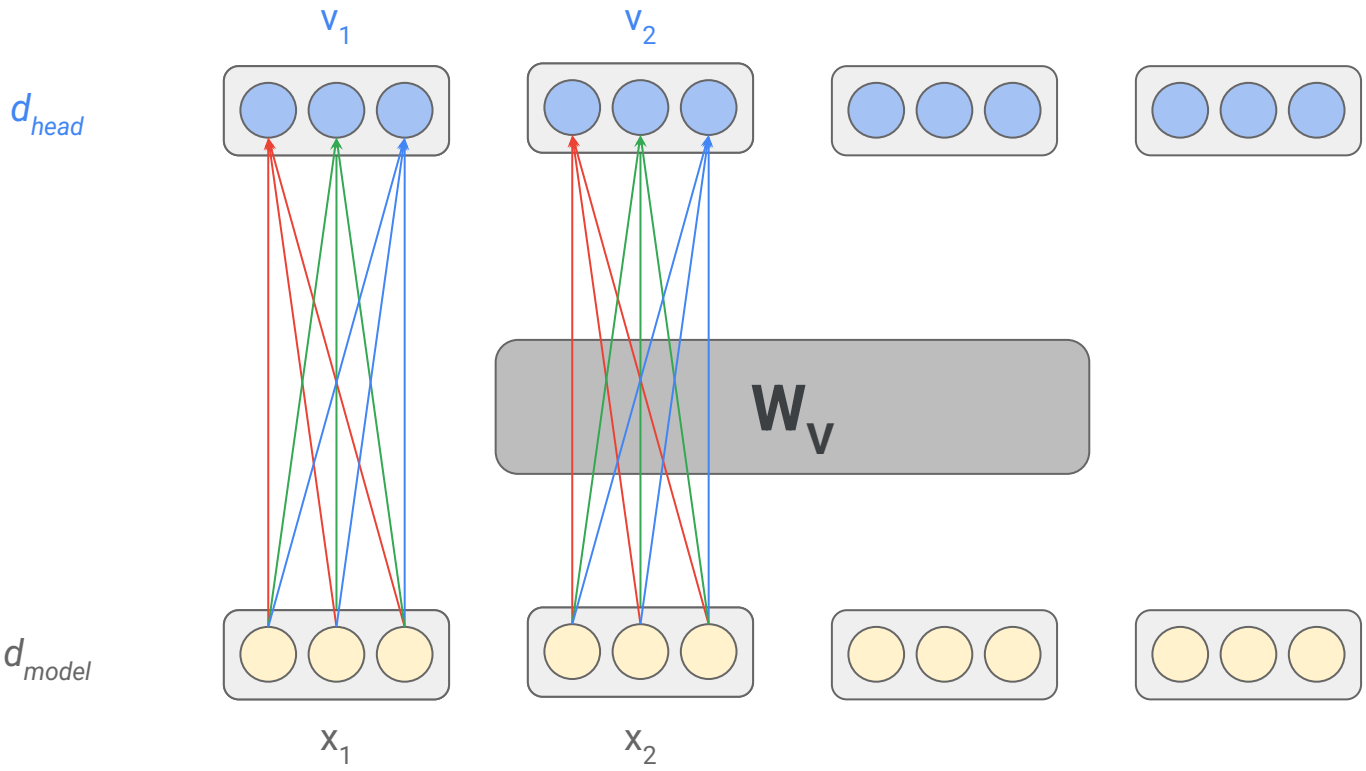
Key vectors



$$K = X \cdot W_K$$

linear projections

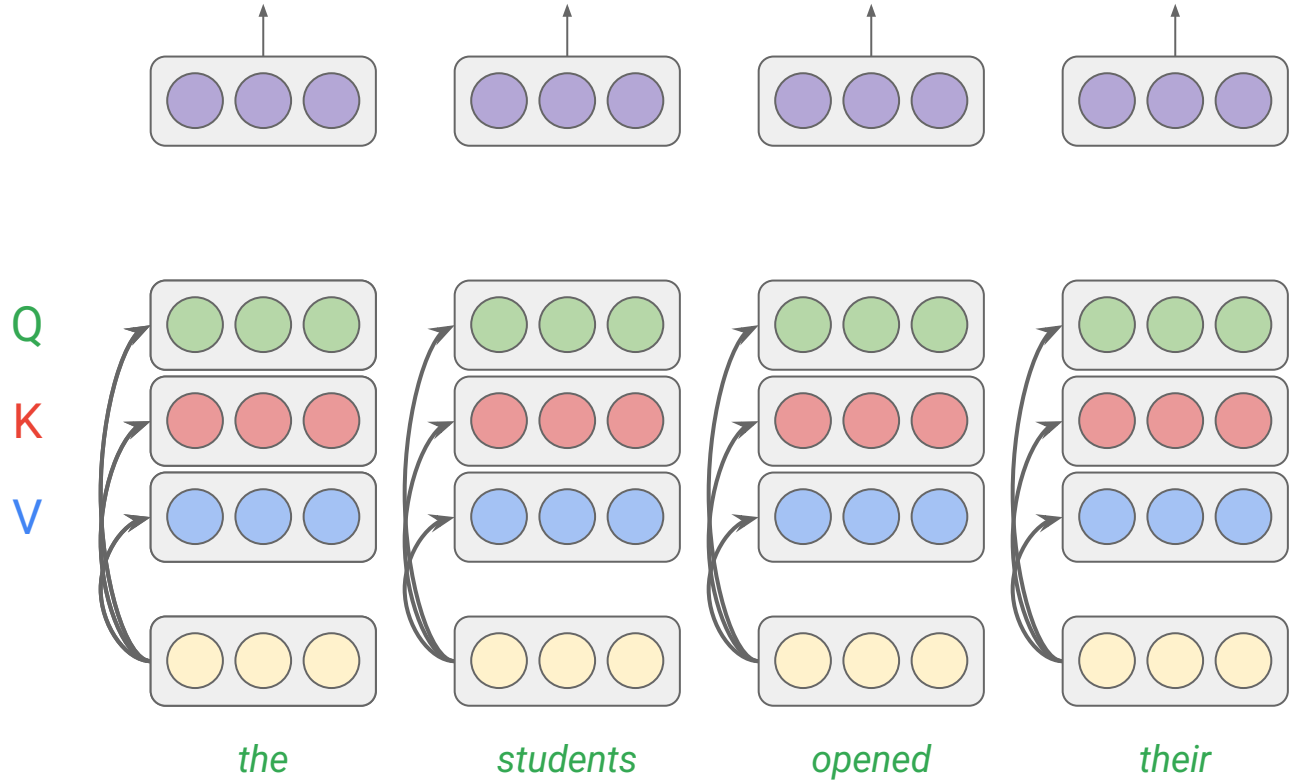
Value vectors



$$V = X \cdot W_V$$

linear projections

Attention (cont'd)



$$Q = X \cdot W_Q$$

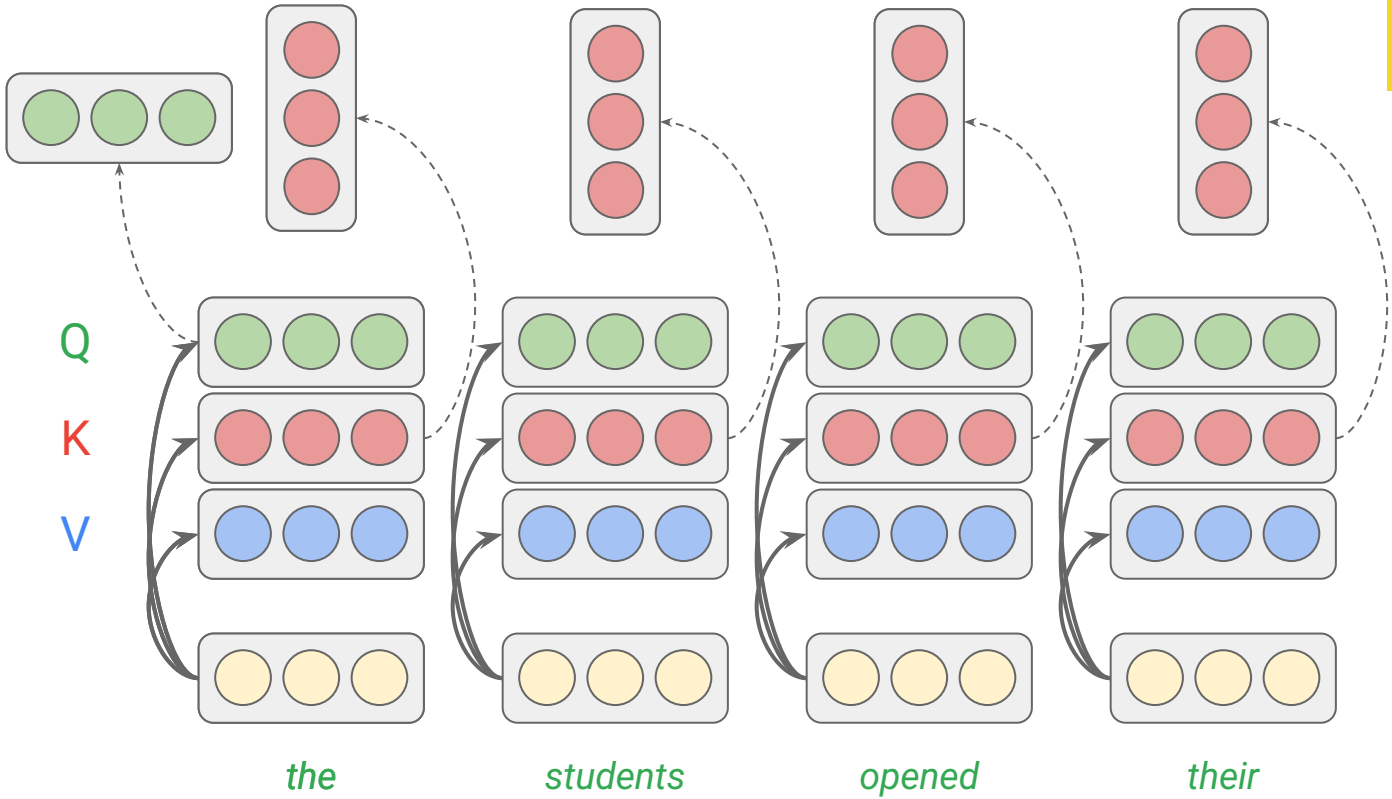
$$K = X \cdot W_K$$

$$V = X \cdot W_V$$

linear projections

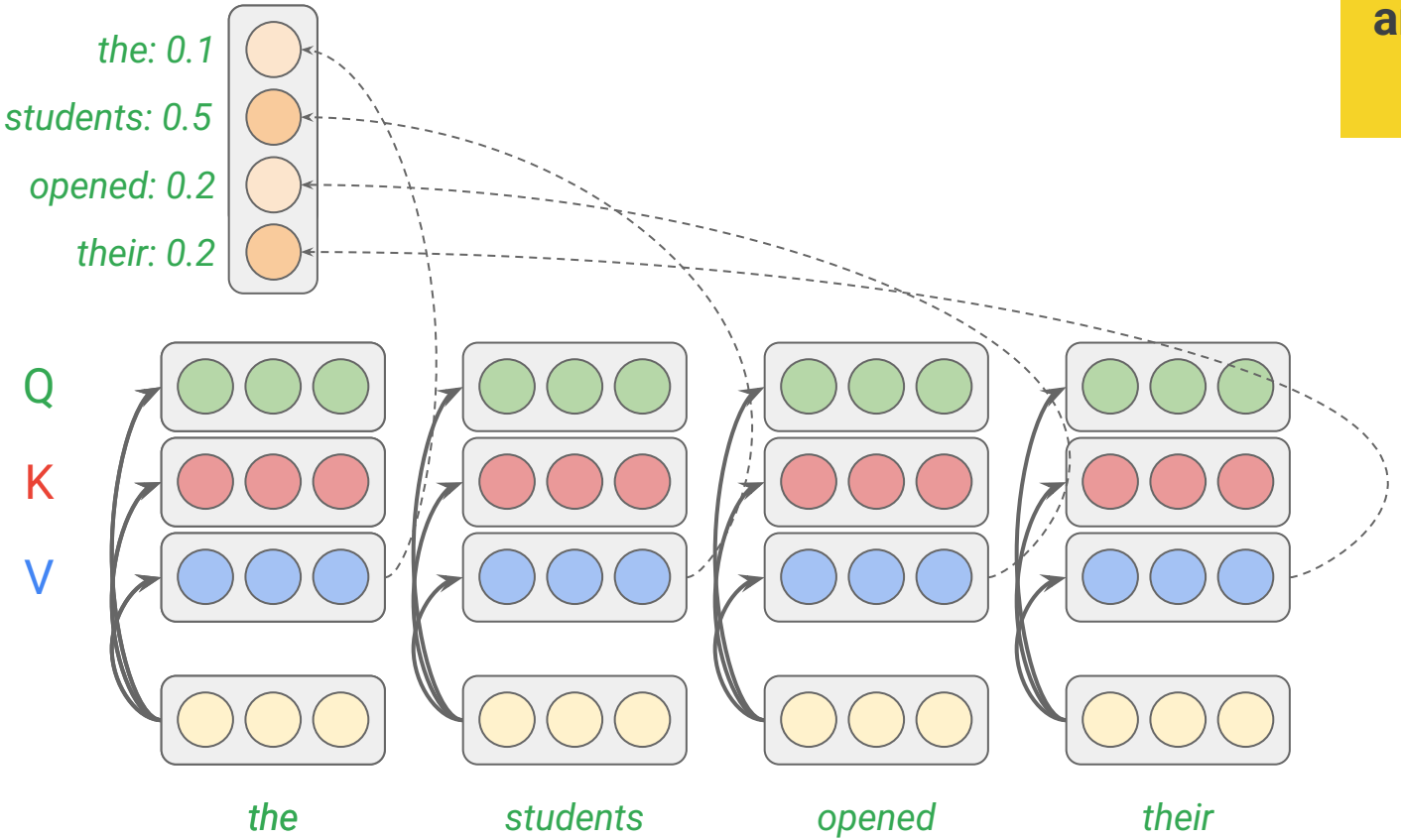
Self-attention (cont'd)

***all* computations are parallelized**



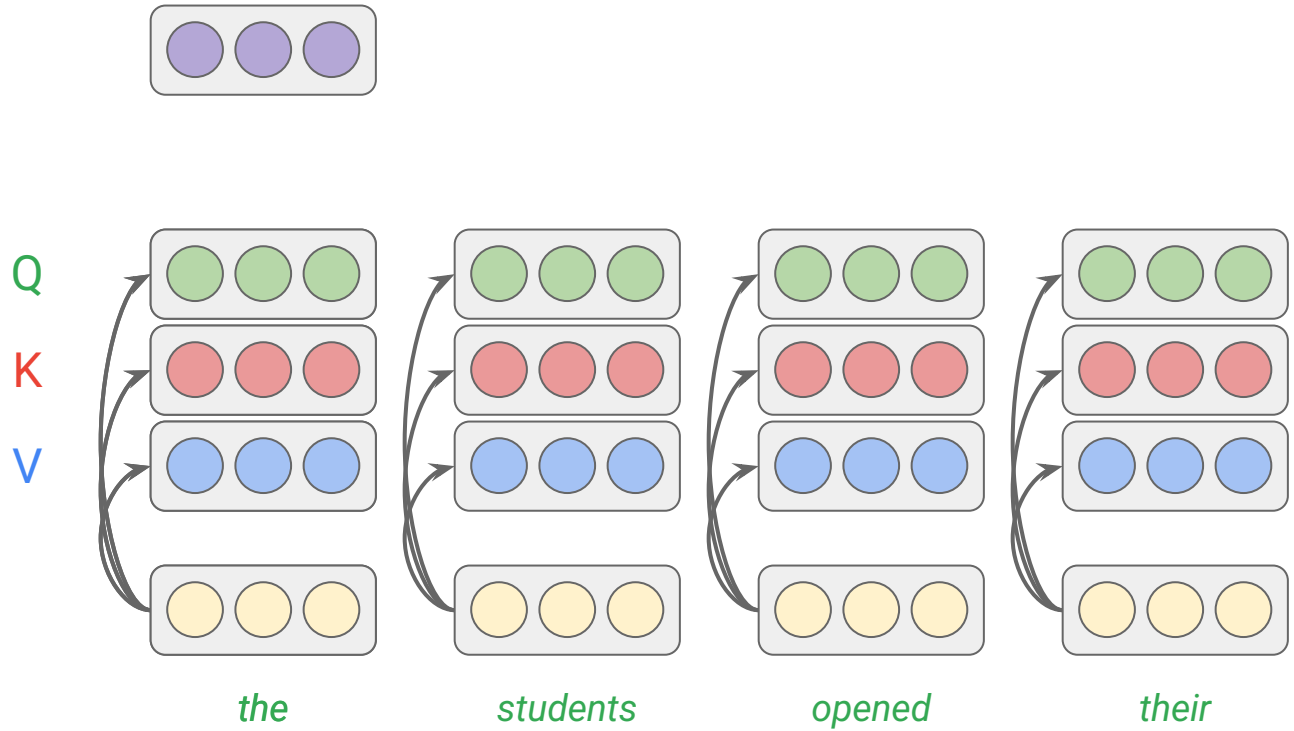
Self-attention (cont'd)

all computations are parallelized

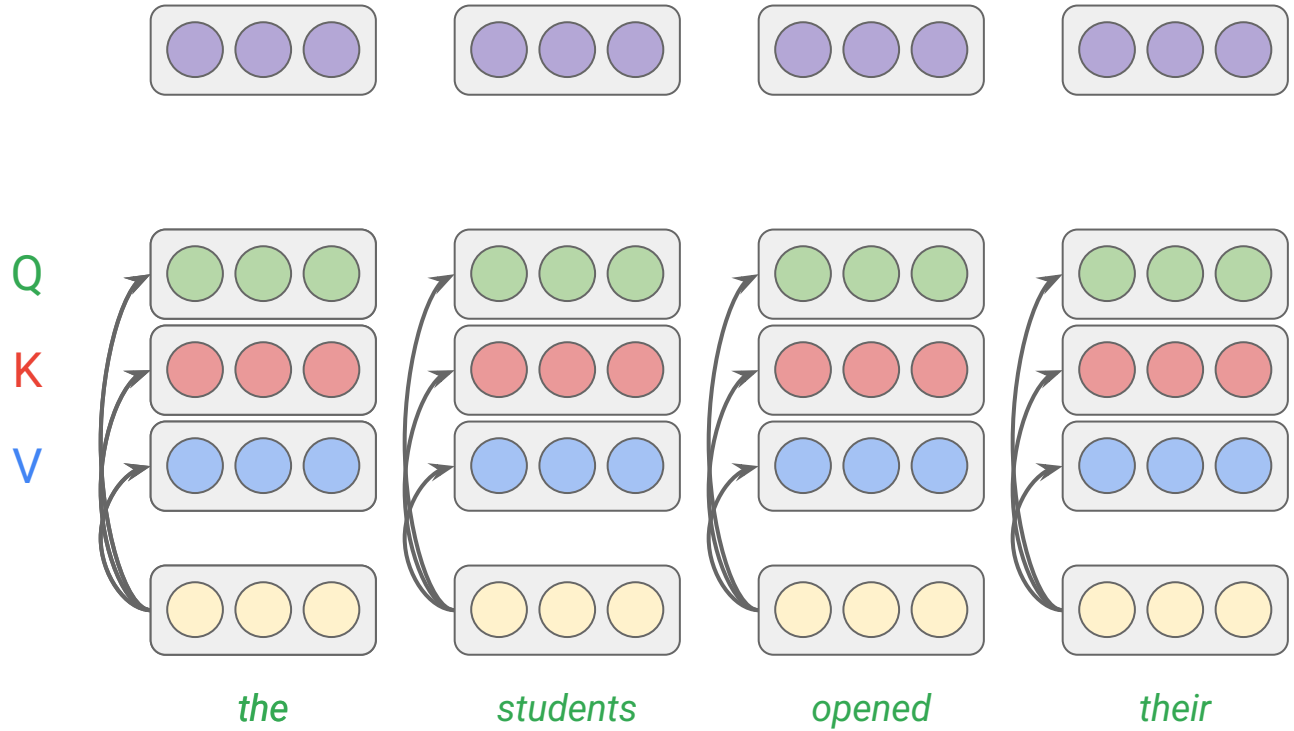


Self-attention (cont'd)

***all* computations are parallelized**

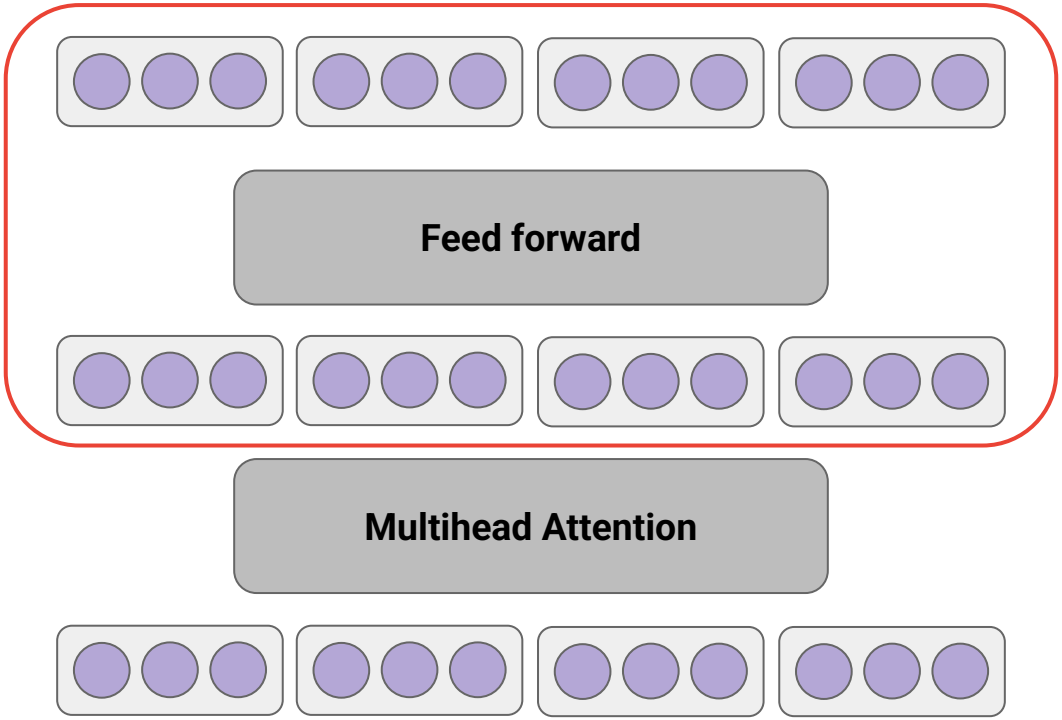


Self-attention (cont'd)

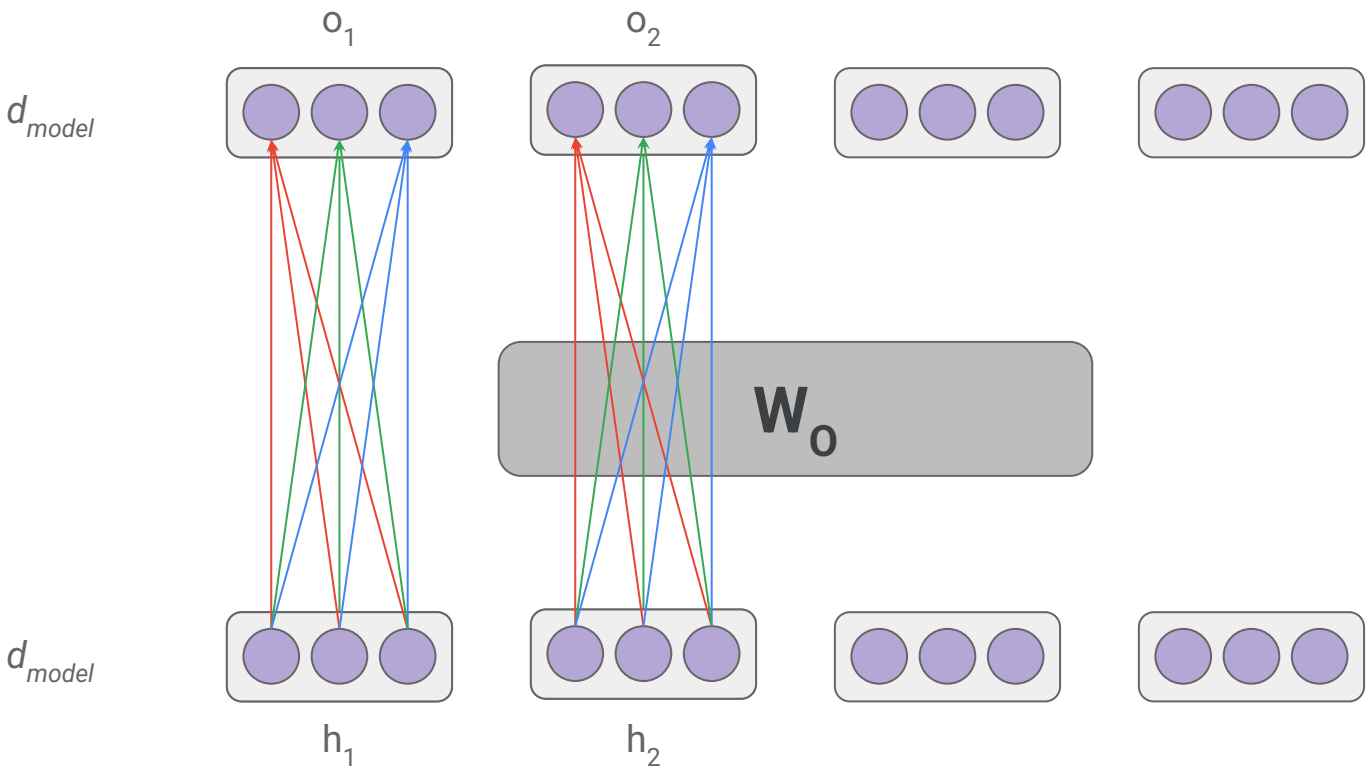


all **computations are parallelized during training and sequential during inference**

Transformer block (cont'd)



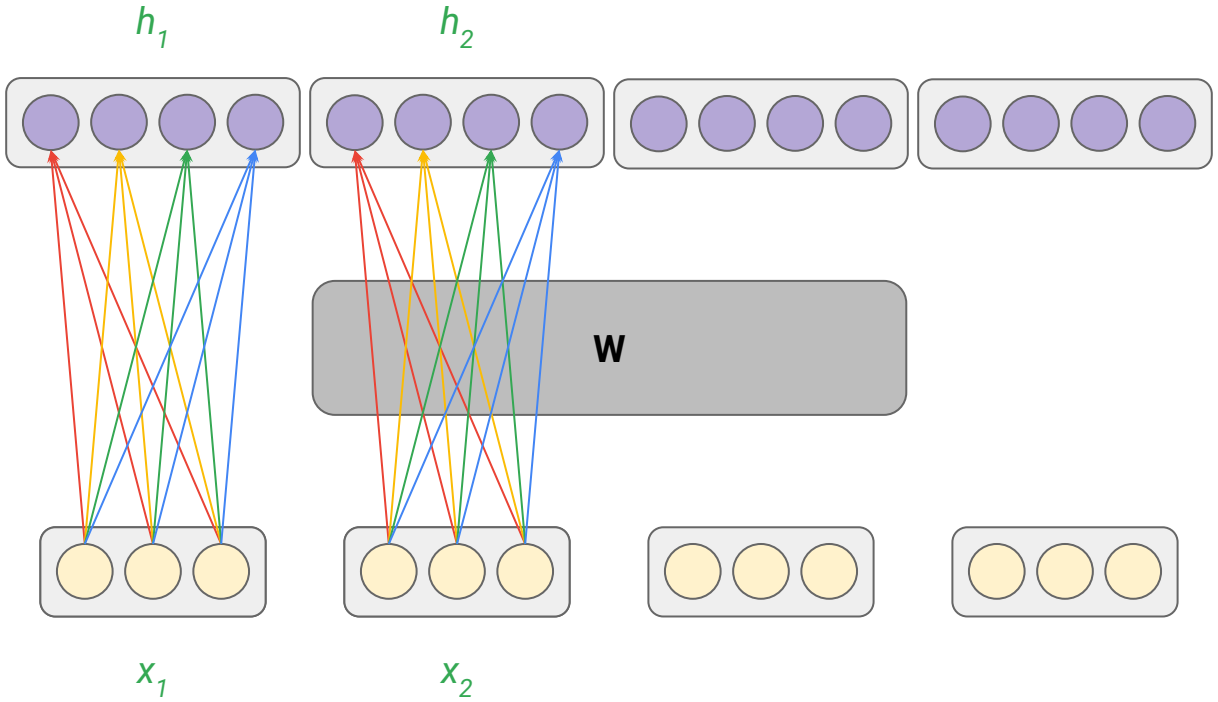
output vectors



$$O = H \cdot W_O$$

linear projections

Position-wise feedforward networks



Thank you!