CS 232: Artificial Intelligence

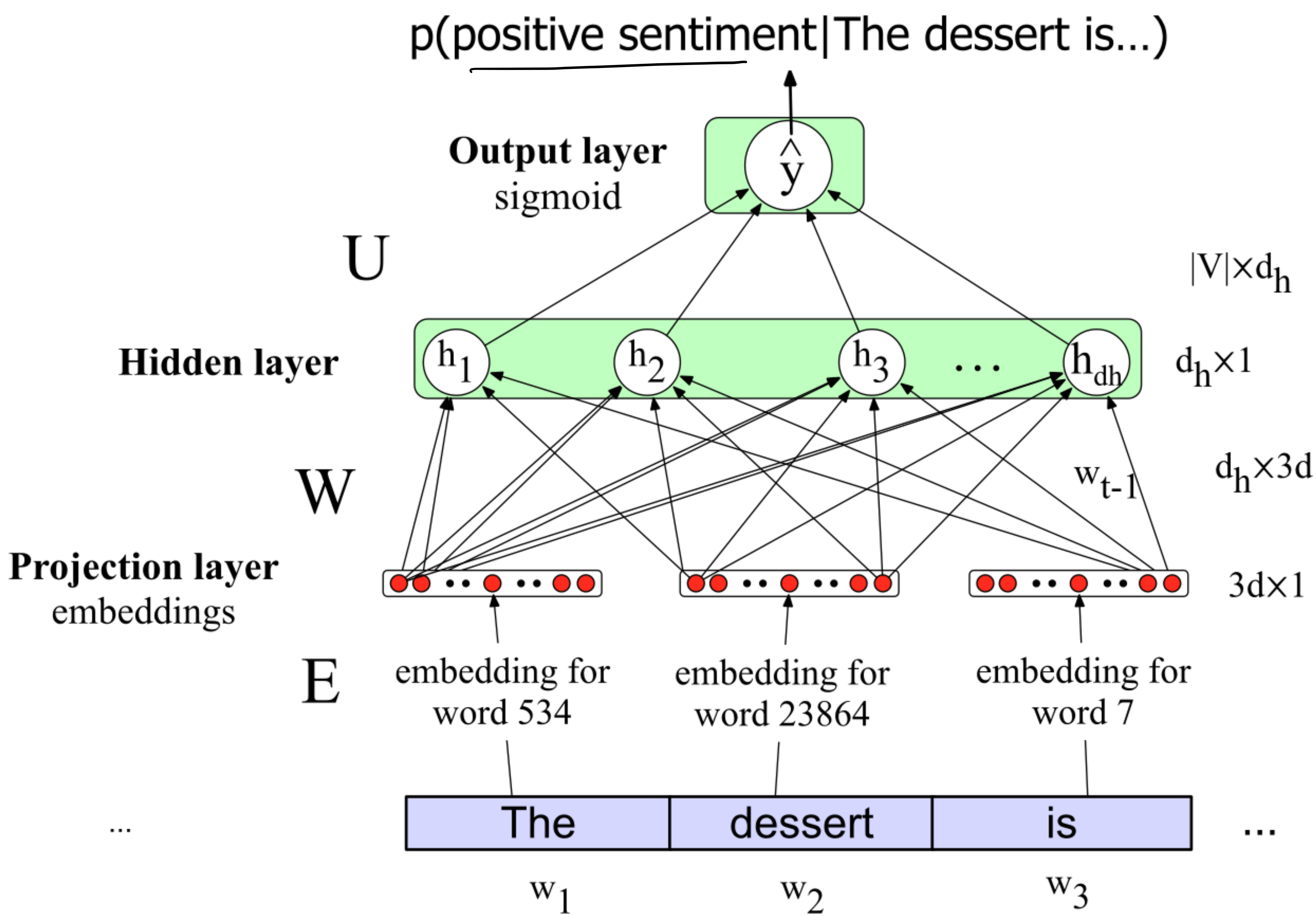
Fall 2023

Prof. Carolyn Anderson
Wellesley College

Reminder

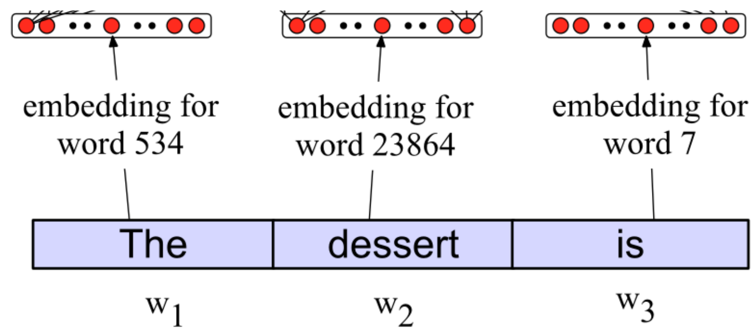
- I'm out of town for a conference most of the week
- No help hours on Thursday!
- Lyra has help hours on Wednesday
- First Gen in CS lunch this Wednesday
- CS Colloquium next Wednesday

Neural Net Classification with embeddings as input features!



Issue: texts come in different sizes

This assumes a fixed size length (3)!



Some simple solutions (more sophisticated solutions later)

1. Make the input the length of the longest review
   - If shorter then pad with zero embeddings
   - Truncate if you get longer reviews at test time
2. Create a single "sentence embedding" (the same dimensionality as a word) to represent all the words
   - Take the mean of all the word embeddings
   - Take the element-wise max of all the word embeddings
     - For each dimension, pick the max value from all words

Recurrent Neural Network

A RNN Language Model

output distribution

$$\hat{y} = \text{softmax}(W_2 h^{(t)} + b_2)$$

hidden states

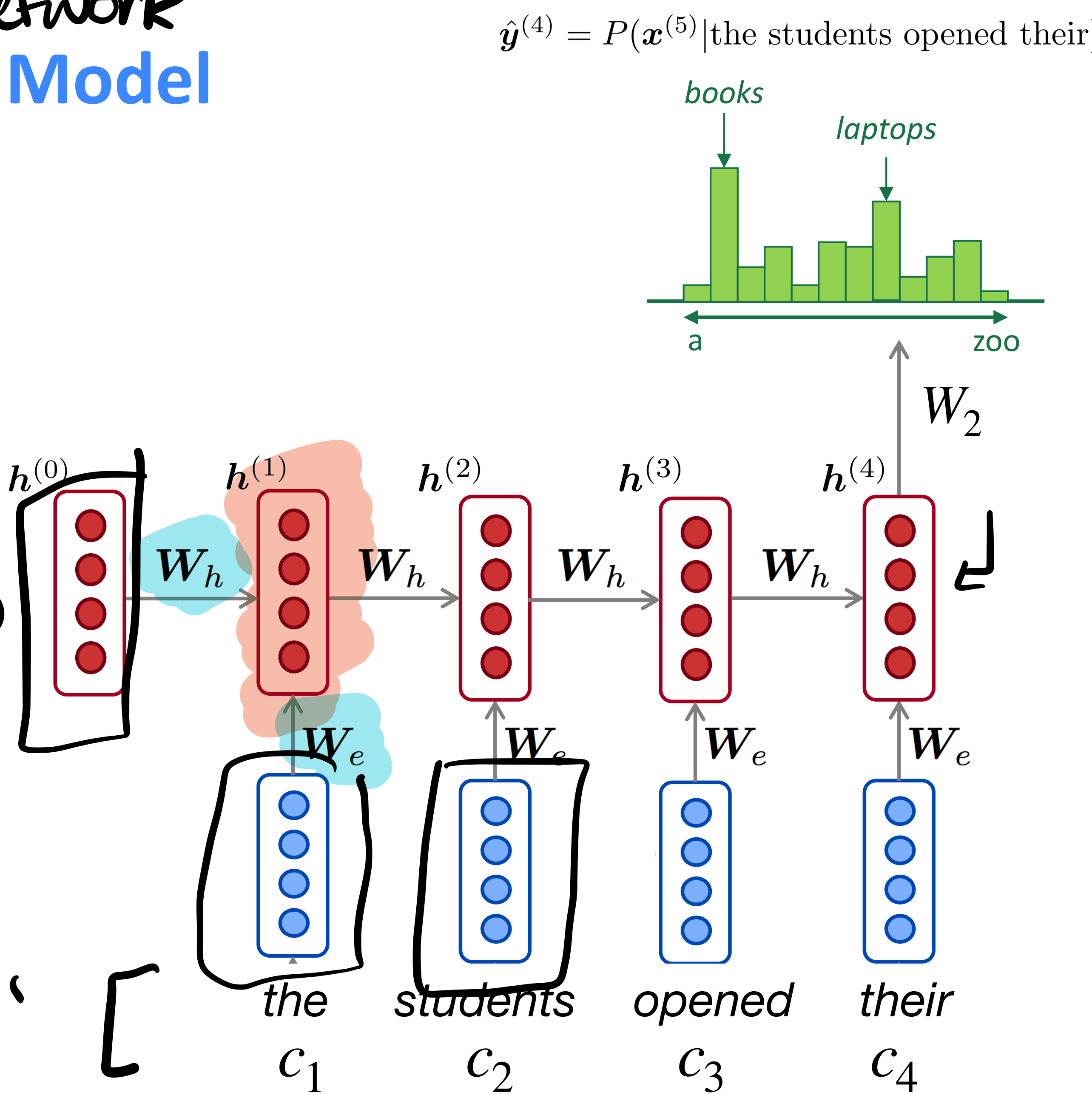
$$h^{(t)} = f(W_h h^{(t-1)} + W_e c_t + b_1)$$

$h^{(0)}$ is initial hidden state!

word embeddings

c_1, c_2, c_3, c_4

timesteps:

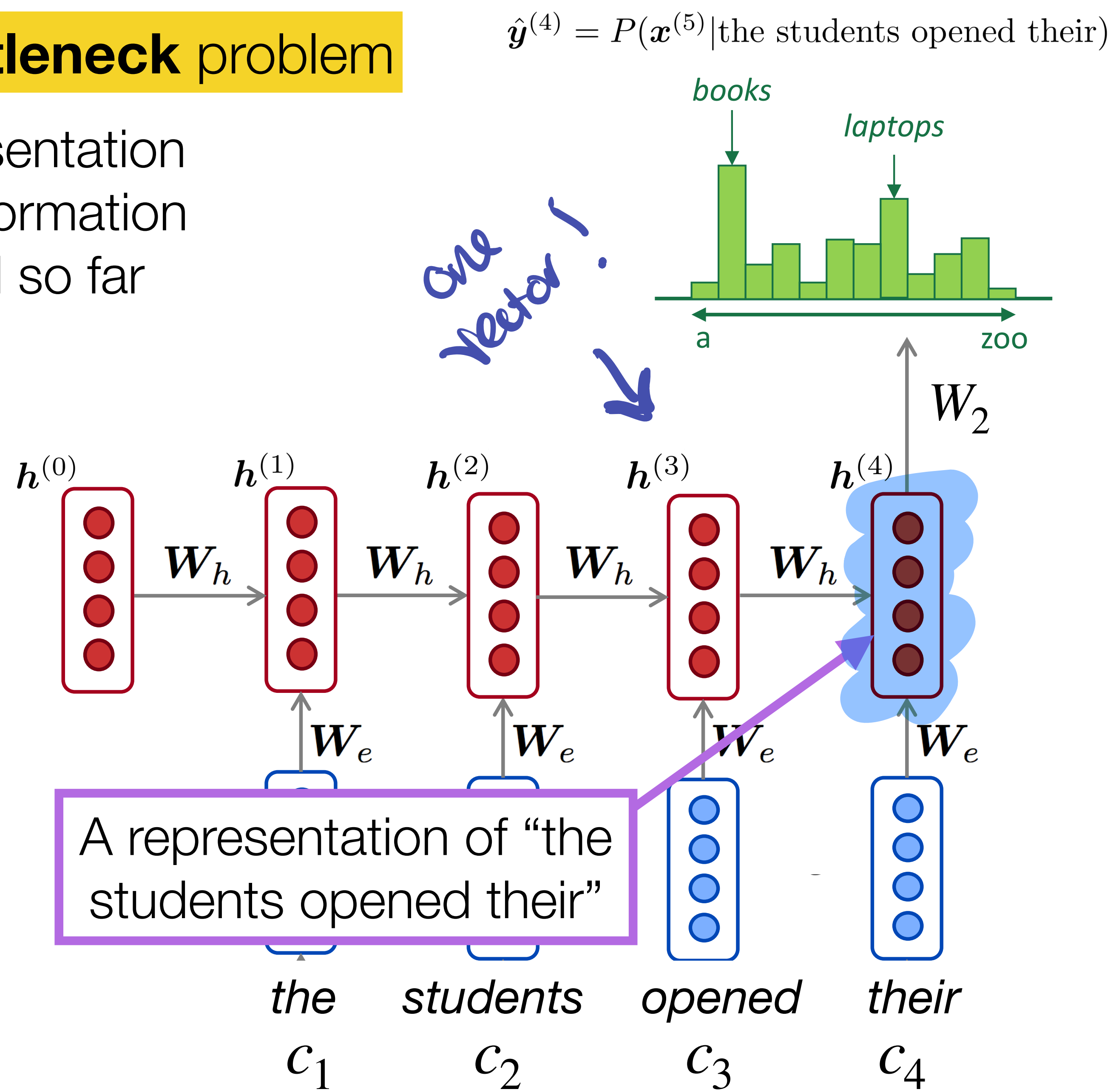


Slides adapted from Mohit Iyyer

RNNs suffer from a **bottleneck** problem

The current hidden representation must encode all of the information about the text observed so far

This becomes difficult especially with longer sequences



Slides adapted from Mohit Iyyer

why is this good?

RNN **Advantages**:

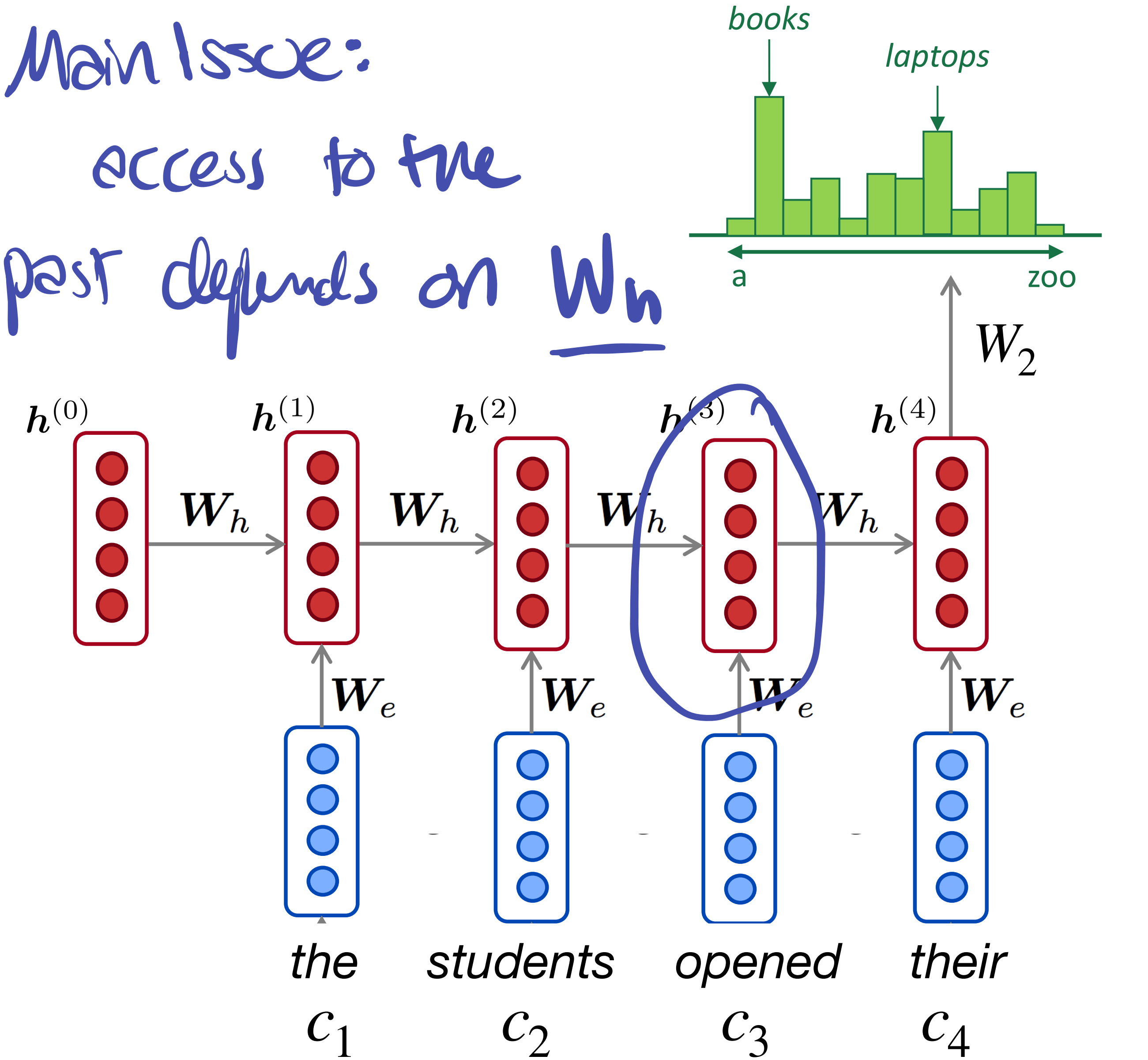
- Can process any length input
- Model size doesn't increase for longer input
- Computation for step *t* can (in theory) use information from many steps back
- Weights are shared across timesteps → representations are shared

RNN **Disadvantages**:

- Recurrent computation is slow
- In practice, difficult to access information from many steps back

$$\hat{\boldsymbol{y}}^{(4)} = P(\boldsymbol{x}^{(5)}|\text{the students opened their})$$

Main Issue:
access to the past depends on W_h



Slides adapted from Mohit Iyyer

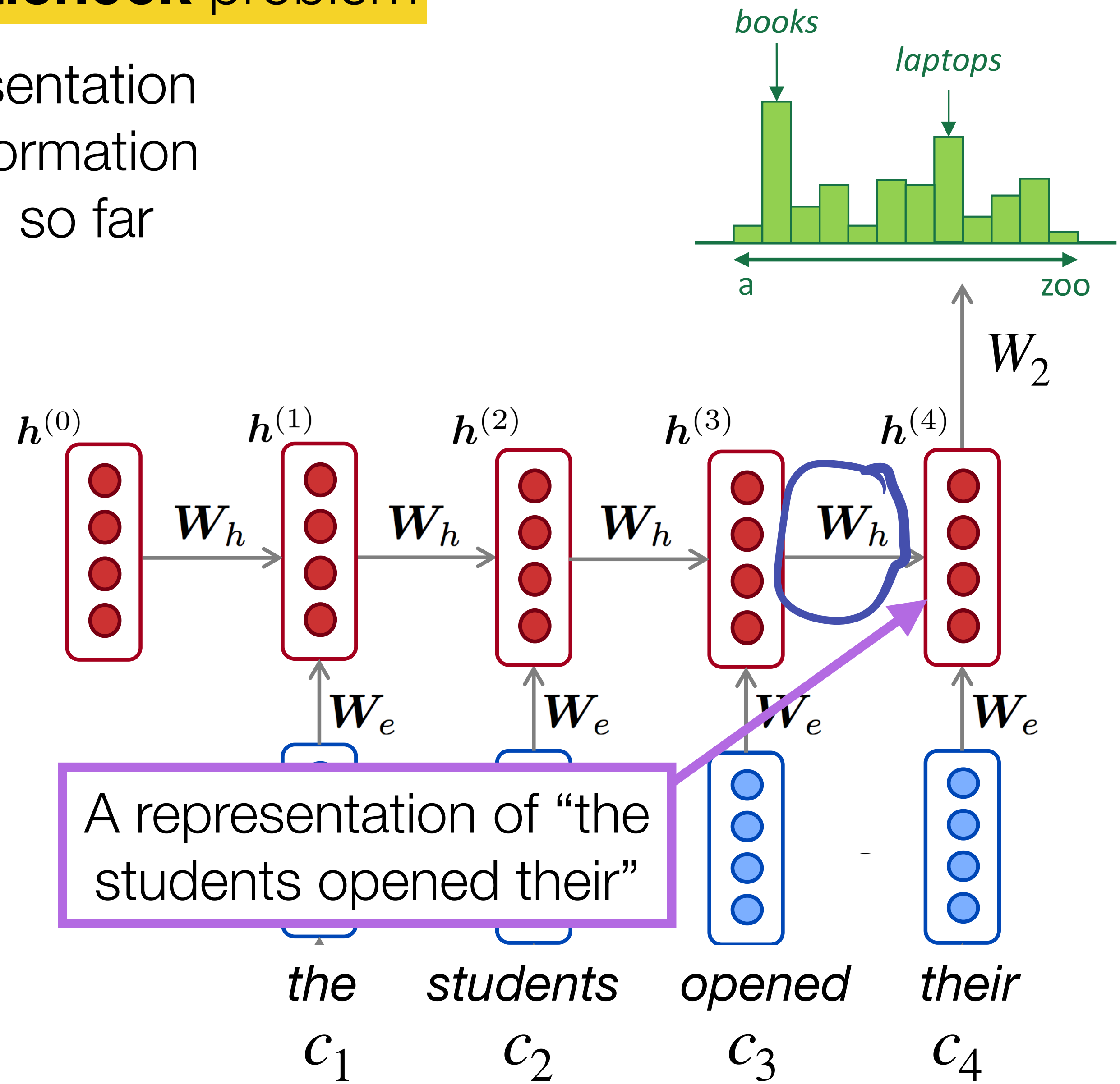
RNNs suffer from a **bottleneck** problem

$$\hat{\boldsymbol{y}}^{(4)} = P(\boldsymbol{x}^{(5)}|\text{the students opened their})$$

The current hidden representation must encode all of the information about the text observed so far

This becomes difficult especially with longer sequences

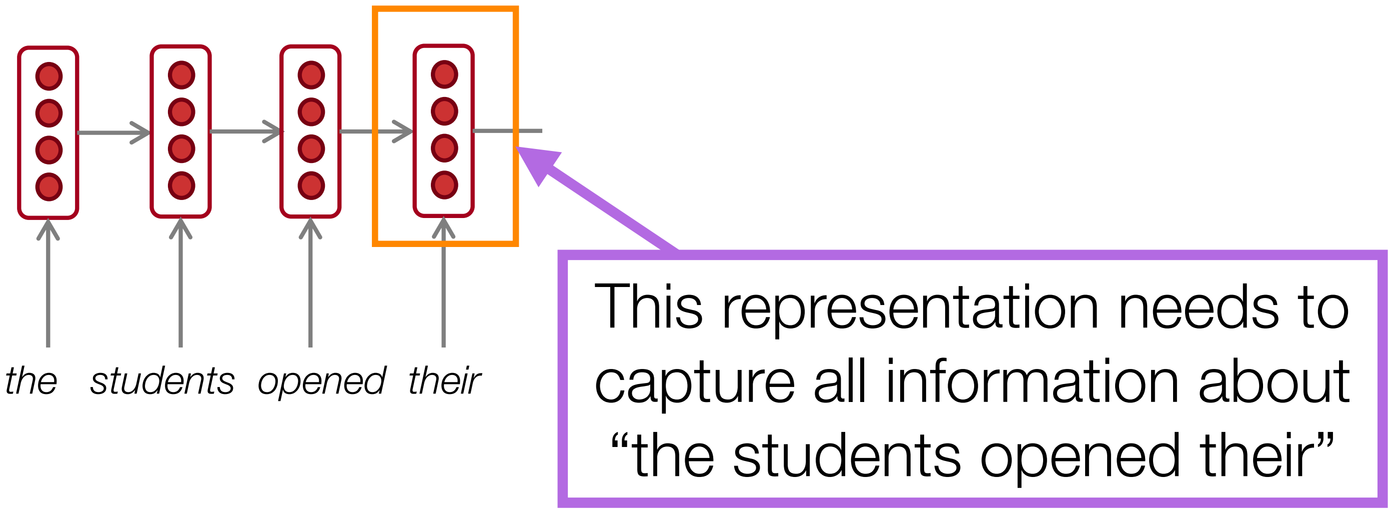
W_h



“you can’t cram the meaning of a whole %&@#&ing sentence into a single $*(&@ing vector!”

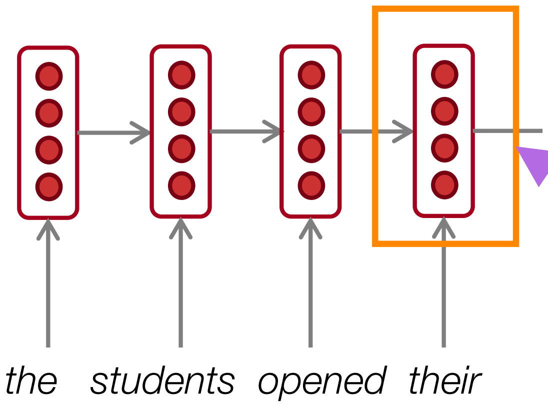
— Ray Mooney (NLP professor at UT Austin)

idea: what if we use multiple vectors?



idea: what if we use multiple vectors?

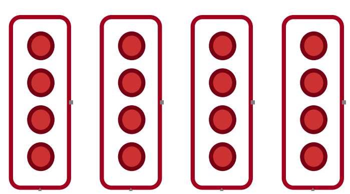
one vector representing the whole past



This representation needs to capture all information about "the students opened their"

multiple vectors representing the past

Instead of this, let's try:

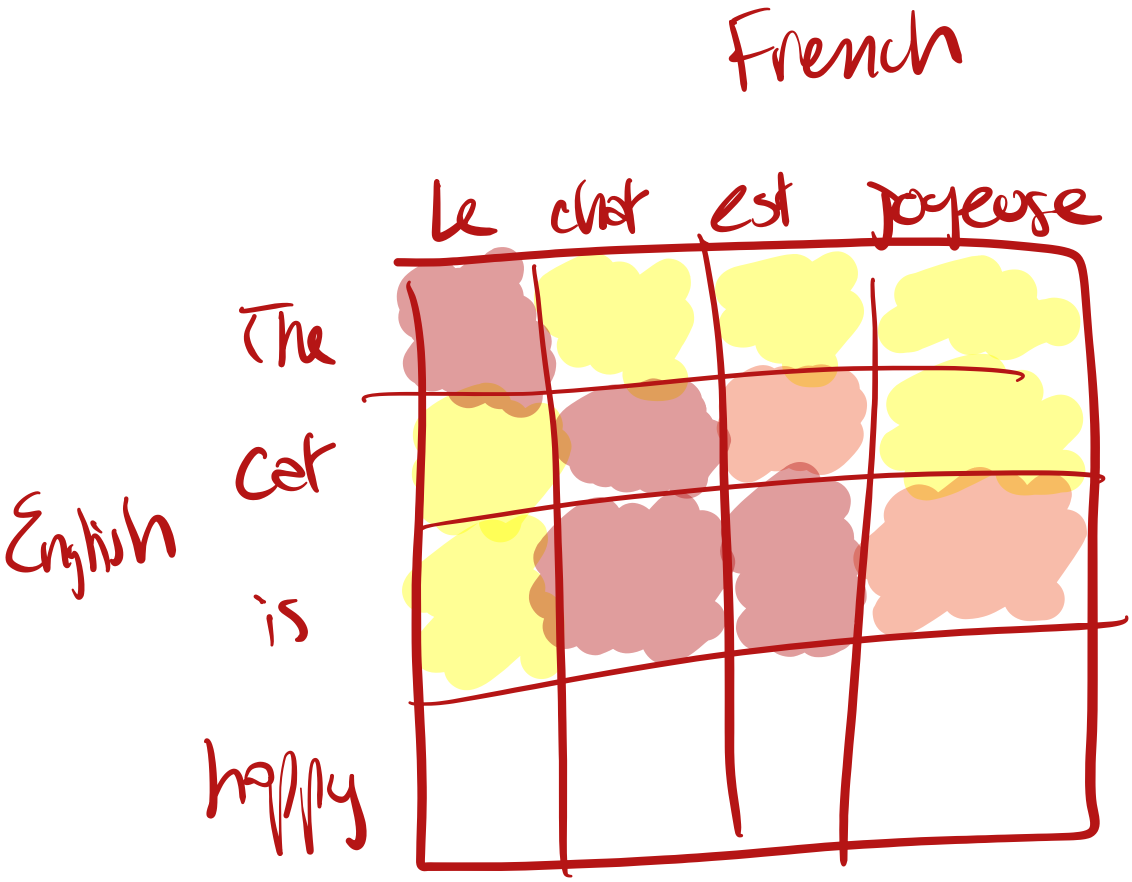
the students opened their =  **(all 4 hidden states!)**

The solution: **attention**

← [Le chat est joyeuse]

The

- **Attention mechanisms** (Bahdanau et al., 2015) allow language models to focus on a particular part of the observed context at each time step
    - Originally developed for machine translation, and intuitively similar to *word alignments* between different languages


French
Le
chat
est
joyeuse
English
The
cat
is
happy

Attention

How does it work?

- in general, we have a single *query* vector and multiple *key* vectors. We want to score each query-key pair

in a neural language model, what are the queries and keys?

What Is Attention?

Goal: learn a task-specific vector v

Intuition: think of v as an "important word" vector

Step 1: Measure the importance of each input vector x by computing its similarity to v. Dot Product

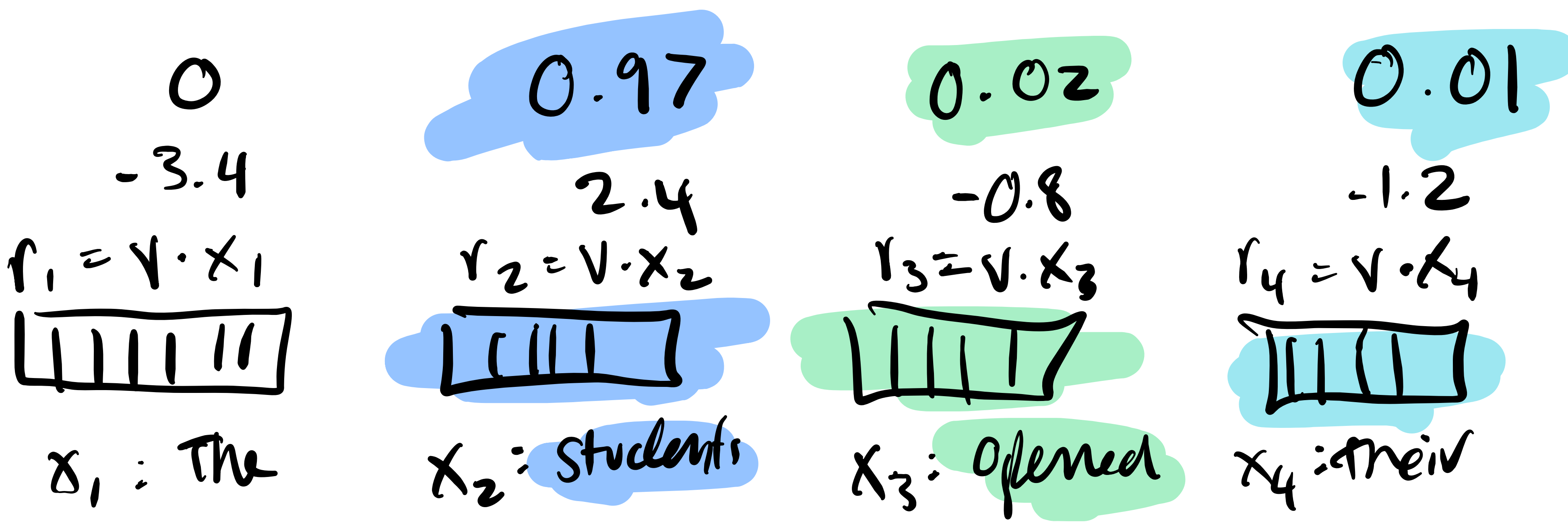
$r_1 = v \cdot x_1$ $r_2 = v \cdot x_2$ $r_3 = v \cdot x_3$ $r_4 = v \cdot x_4$

x_1 : The x_2 : students x_3 : opened x_4 : their

What Is Attention?

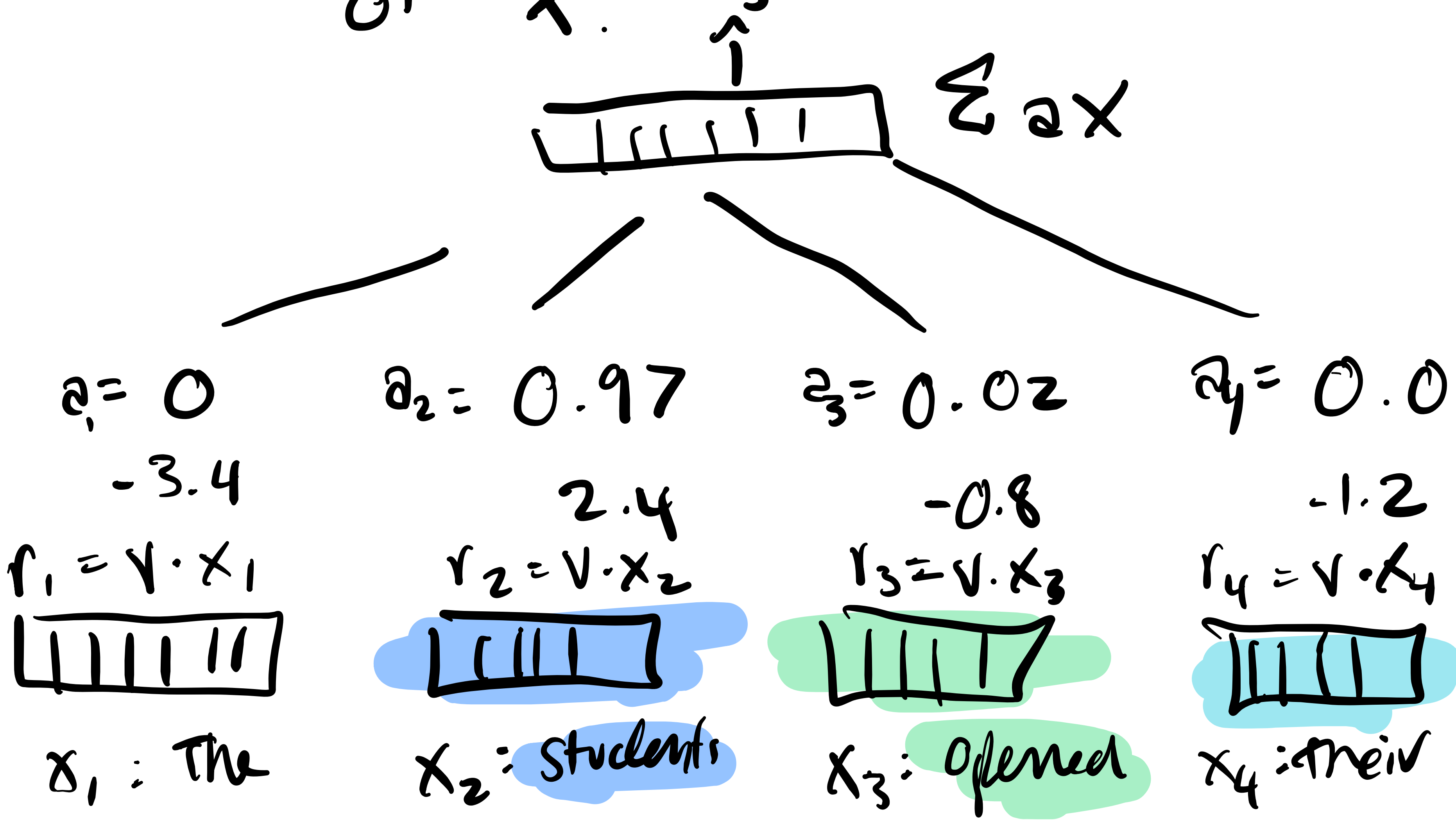
Step 2: Take r & normalize it using softmax

$$a = \mathrm{softmax}(r)$$

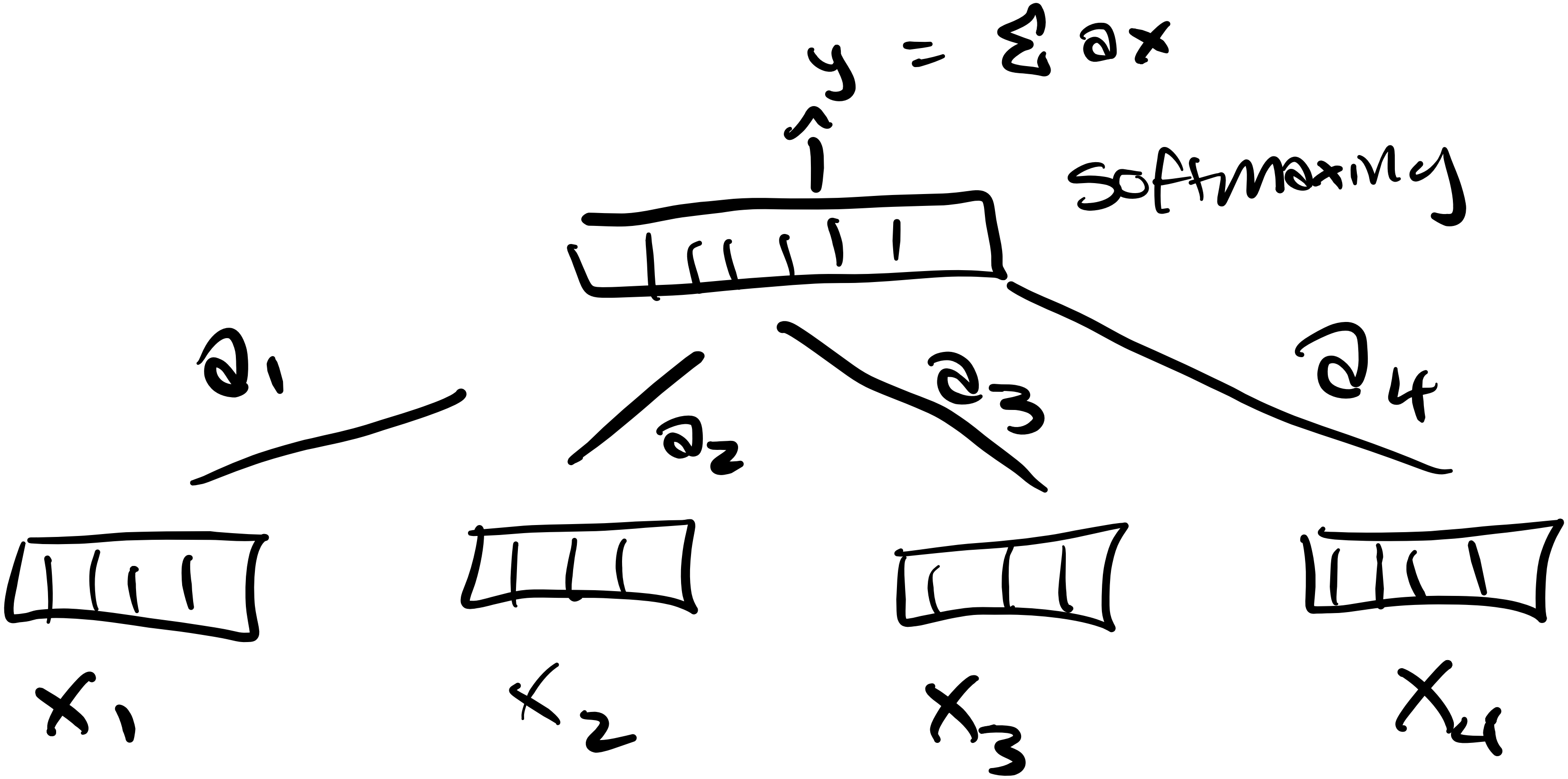


What Is Attention?

Step 3: Compute a weighted average of X. $y = \sum ax$



What Is Attention?



$$y = \mathrm{softmax}(ax)$$

Vicki
@vboykis

They don’t tell you this in the paper (well they do but you have to read it like 15 times)

6:20 PM · Feb 22, 2023 · **88.1K** Views

Why dot product?

- Dot product provides a measure of similarity between keys and queries.
- But you might be wondering: *why do we want to pay attention to words that are similar to the current word*?

Why dot product?

- Dot product provides a measure of similarity between keys and queries.

- But you might be wondering: *why do we want to pay attention to words that are similar to the current word*?

Consider:

My brother, a chemist, was late yesterday because he missed the bus. When he arrived, he was surprised to find that his lab ________

Why dot product?

- Dot product provides a measure of similarity between keys and queries.
- But you might be wondering: *why do we want to pay attention to words that are similar to the current word*?

Consider:

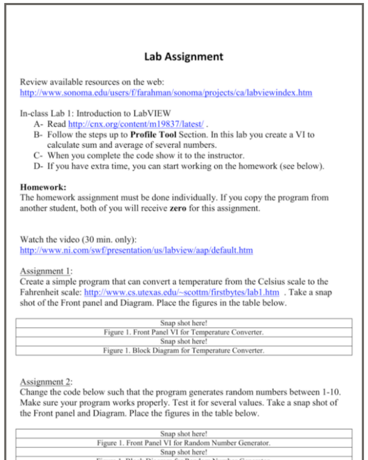
My brother, a chemist, was late yesterday because he missed the bus. When he arrived, he was surprised to find that his lab ________

lab

lab

lab

lab



Lab Assignment

Review available resources on the web:
http://www.sonoma.edu/users/f/farahman/sonoma/projects/ca/labviewindex.htm

In-class Lab 1: Introduction to LabVIEW
A- Read http://cnx.org/content/m19837/latest/ .
B- Follow the steps up to **Profile Tool** Section. In this lab you create a VI to calculate sum and average of several numbers.
C- When you complete the code show it to the instructor.
D- If you have extra time, you can start working on the homework (see below).

Homework:
The homework assignment must be done individually. If you copy the program from another student, both of you will receive **zero** for this assignment.

Watch the video (30 min. only):
http://www.ni.com/swf/presentation/us/labview/aap/default.htm

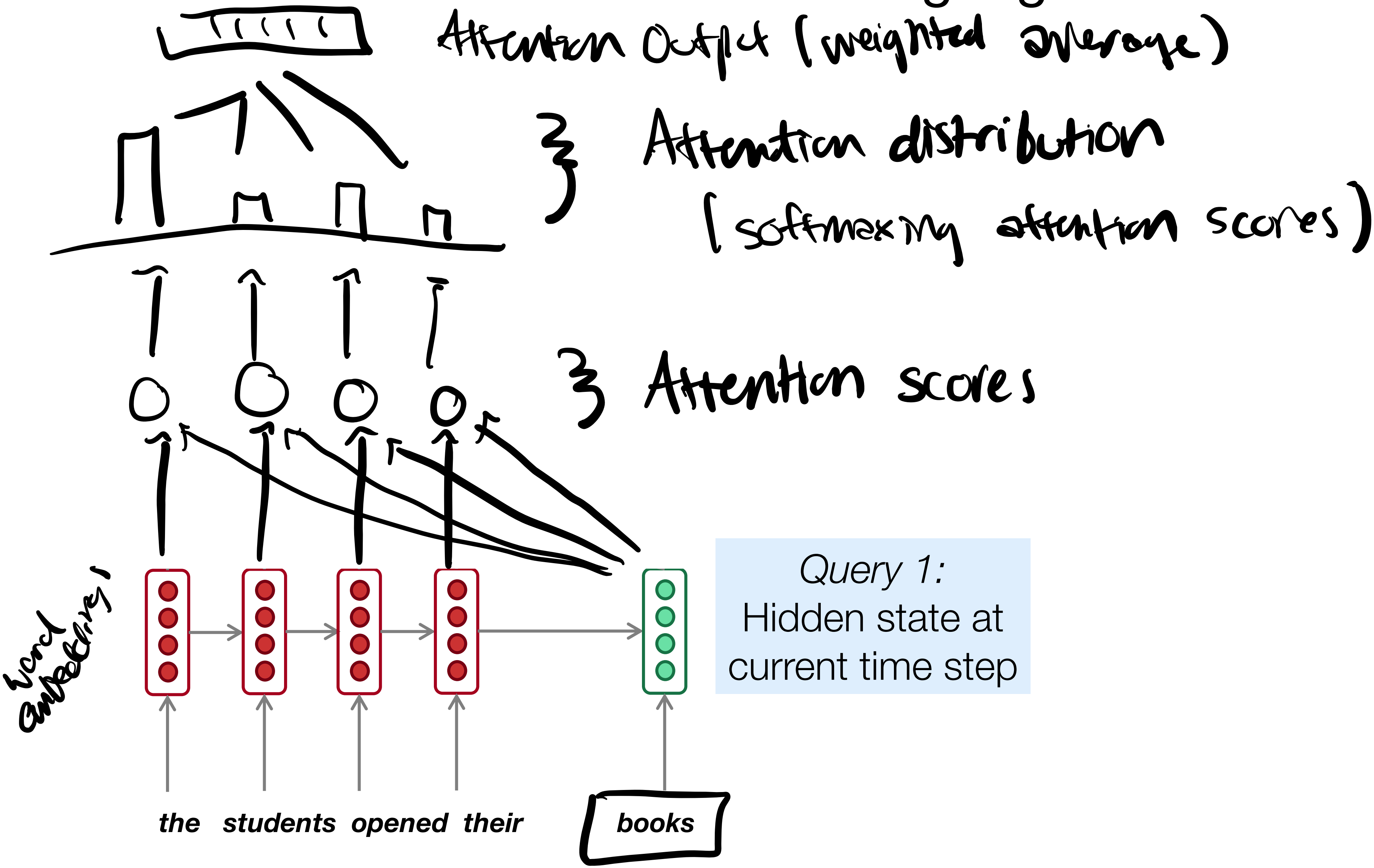
Assignment 1:
Create a simple program that can convert a temperature from the Celsius scale to the Fahrenheit scale: http://www.cs.utexas.edu/~scottm/firstbytes/lab1.htm . Take a snap shot of the Front panel and Diagram. Place the figures in the table below.

Snap shot here!
Figure 1. Front Panel VI for Temperature Converter.
Snap shot here!
Figure 1. Block Diagram for Temperature Converter.

Assignment 2:
Change the code below such that the program generates random numbers between 1-10. Make sure your program works properly. Test it for several values. Take a snap shot of the Front panel and Diagram. Place the figures in the table below.

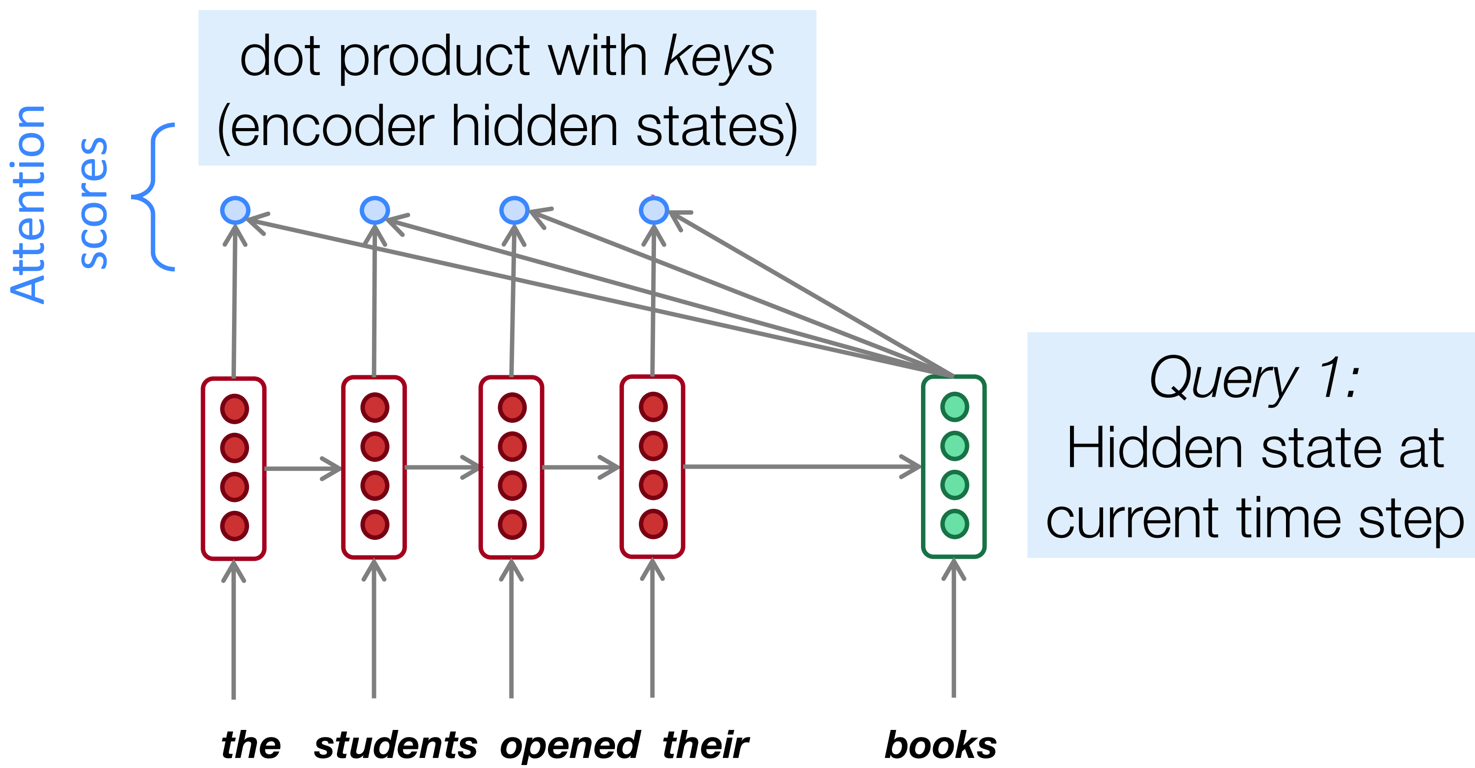
Snap shot here!
Figure 1. Front Panel VI for Random Number Generator.
Snap shot here!

Attention mechanisms in neural language models



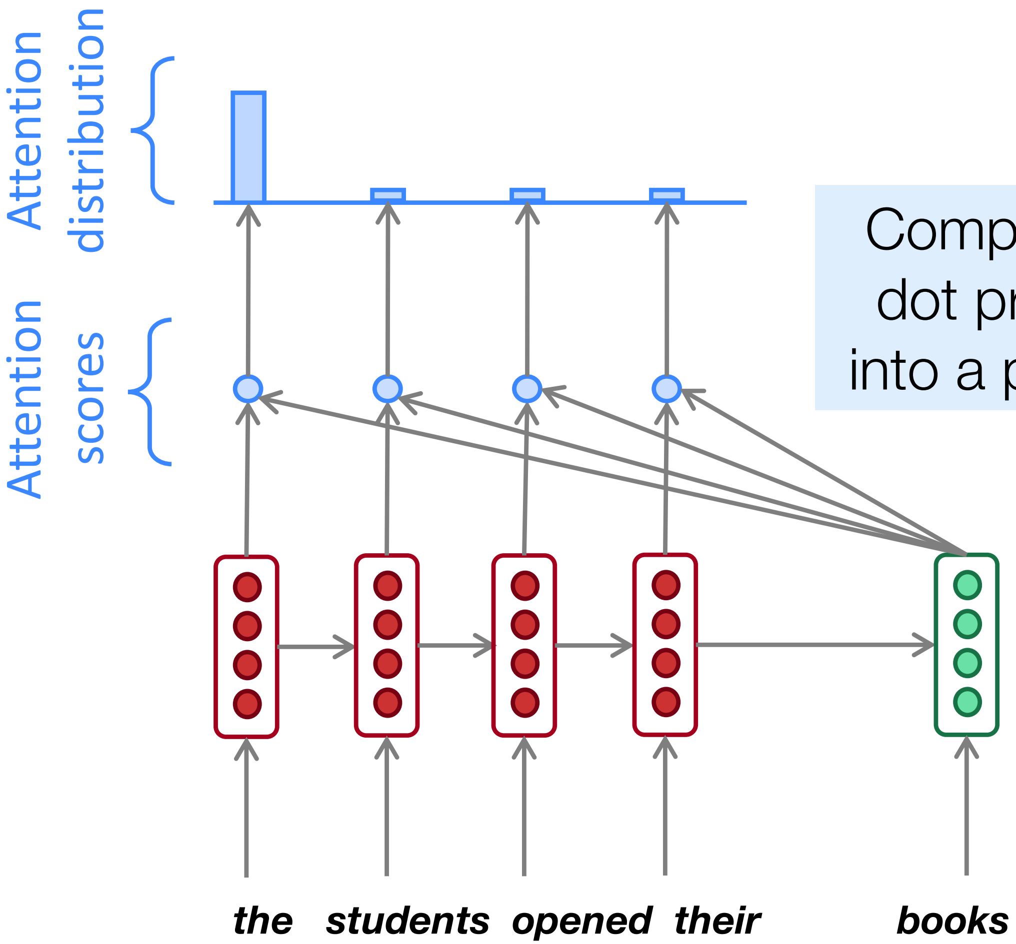
Slides adapted from Mohit Iyyer

Attention mechanisms in neural language models



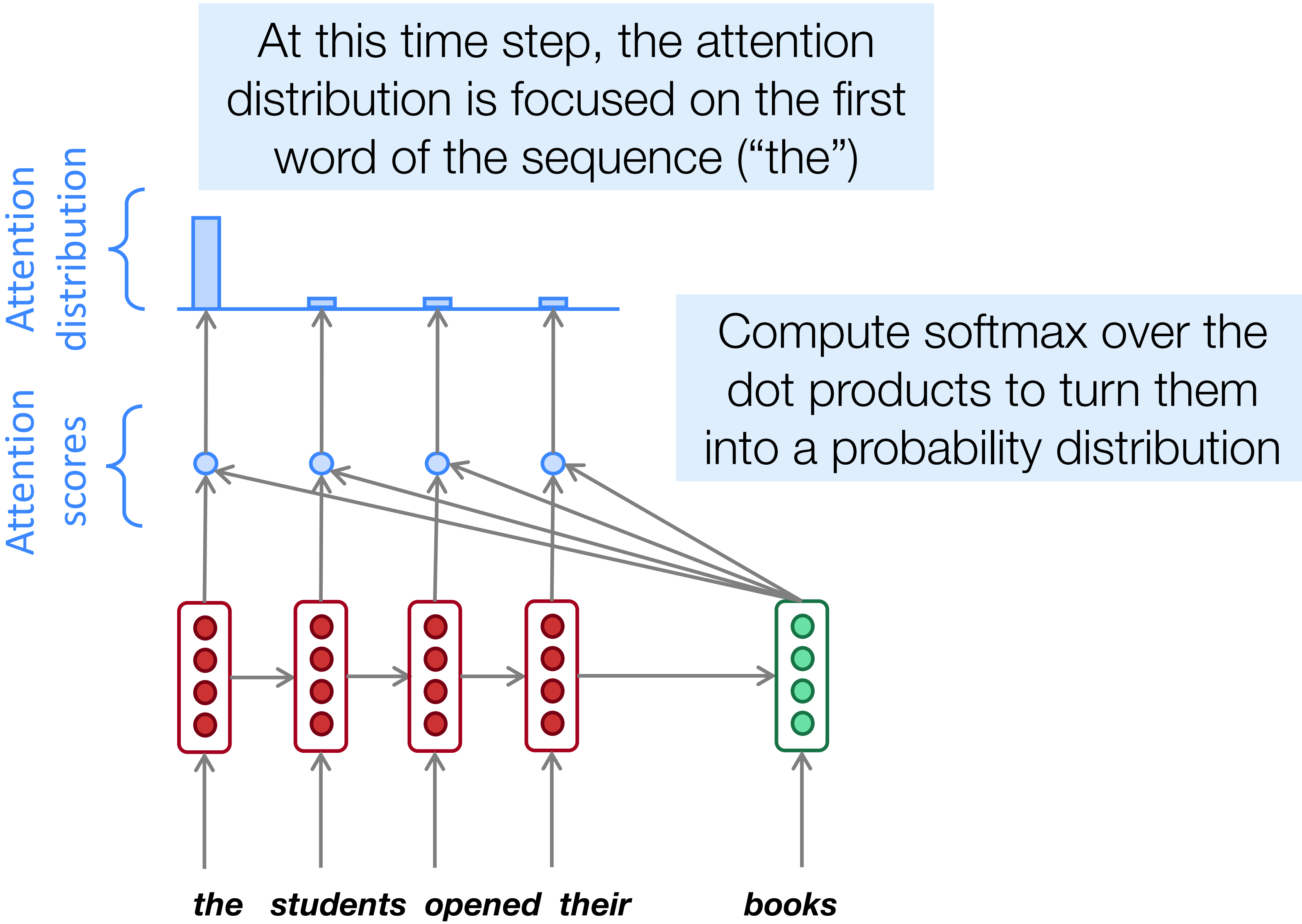
Slides adapted from Mohit Iyyer

Attention mechanisms in neural language models



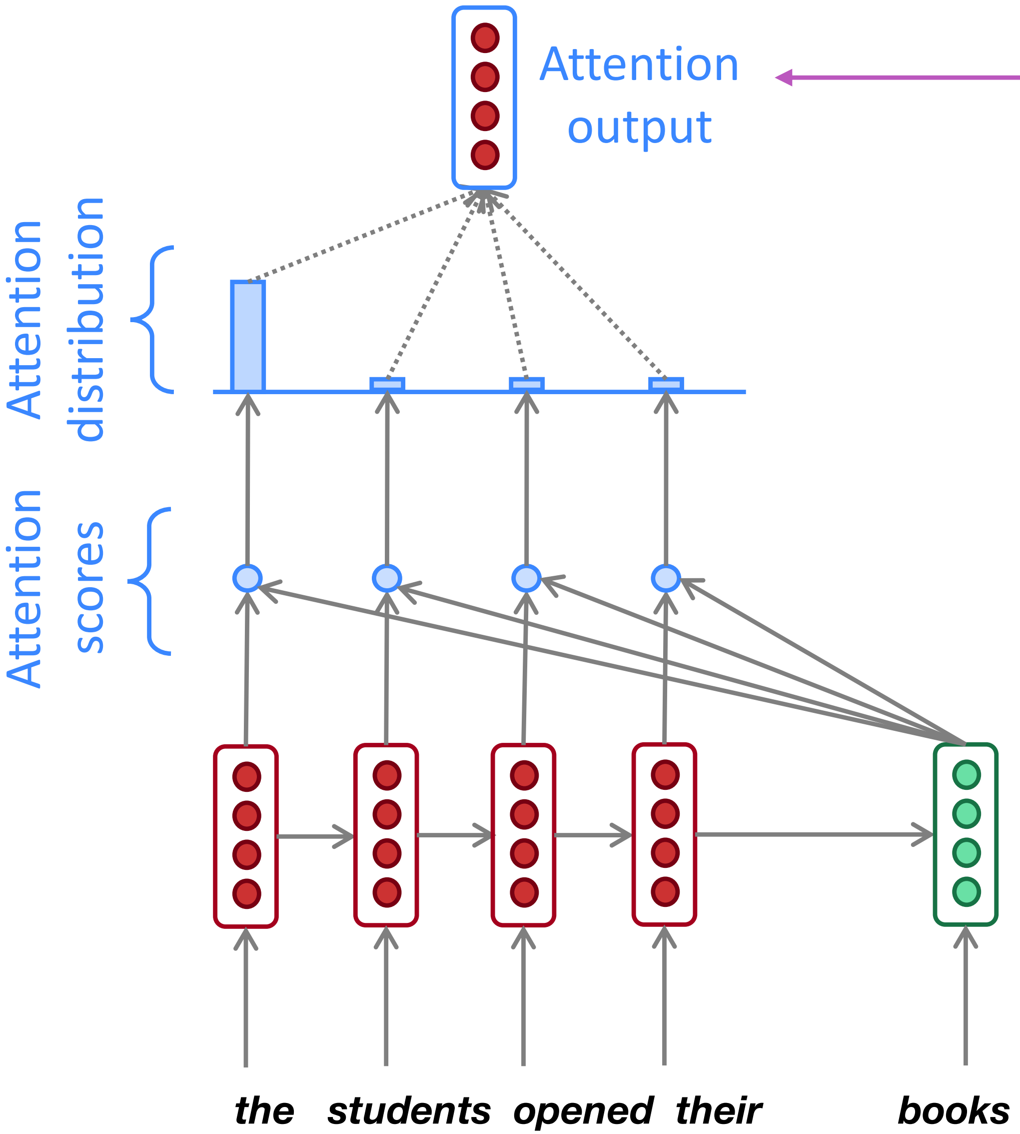
Compute softmax over the dot products to turn them into a probability distribution

Attention mechanisms in neural language models



Slides adapted from Mohit Iyyer

Attention mechanisms in neural language models

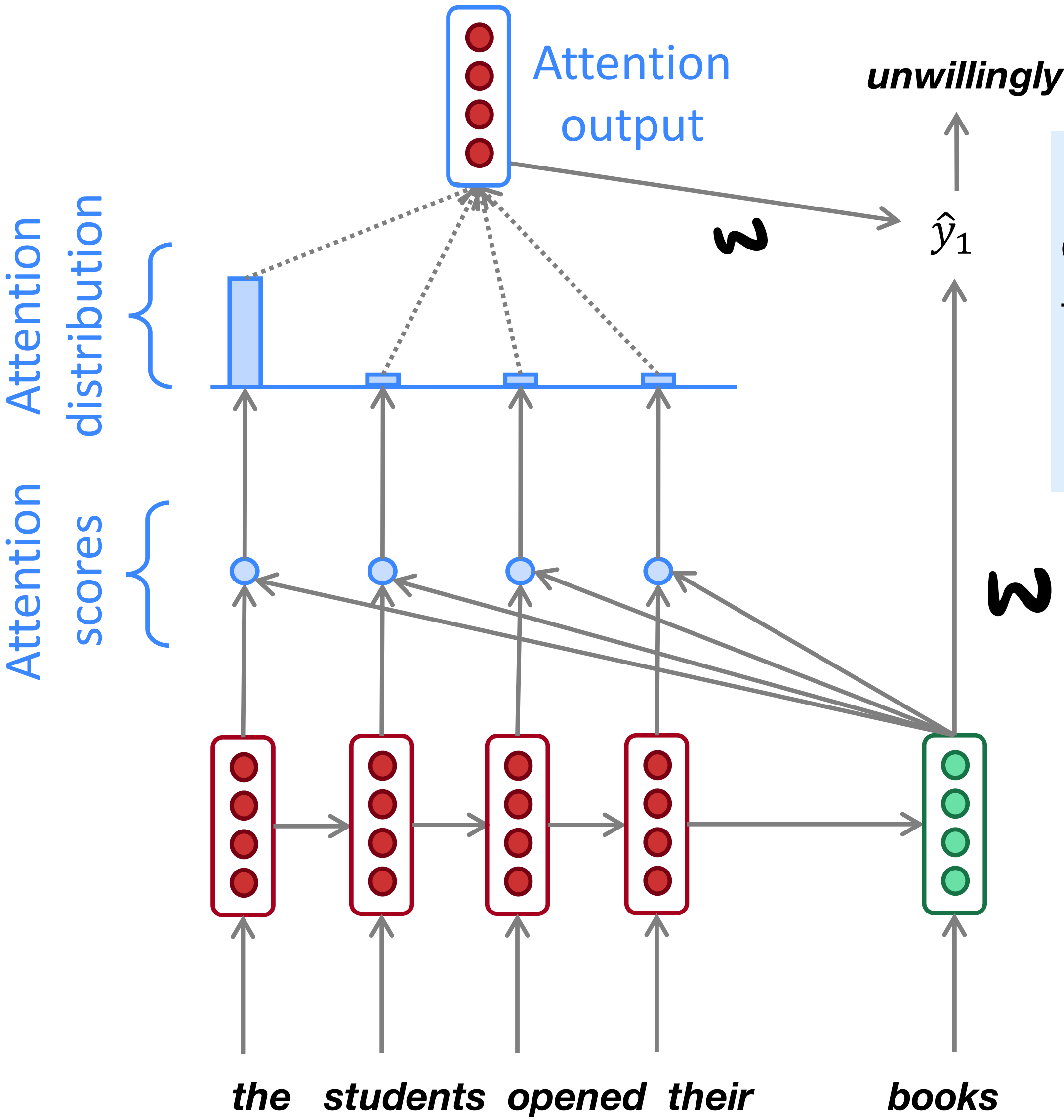


We use the attention distribution to compute a weighted average of the hidden states.

Intuitively, the resulting attention output contains information from hidden states that received high attention scores

Slides adapted from Mohit Iyyer

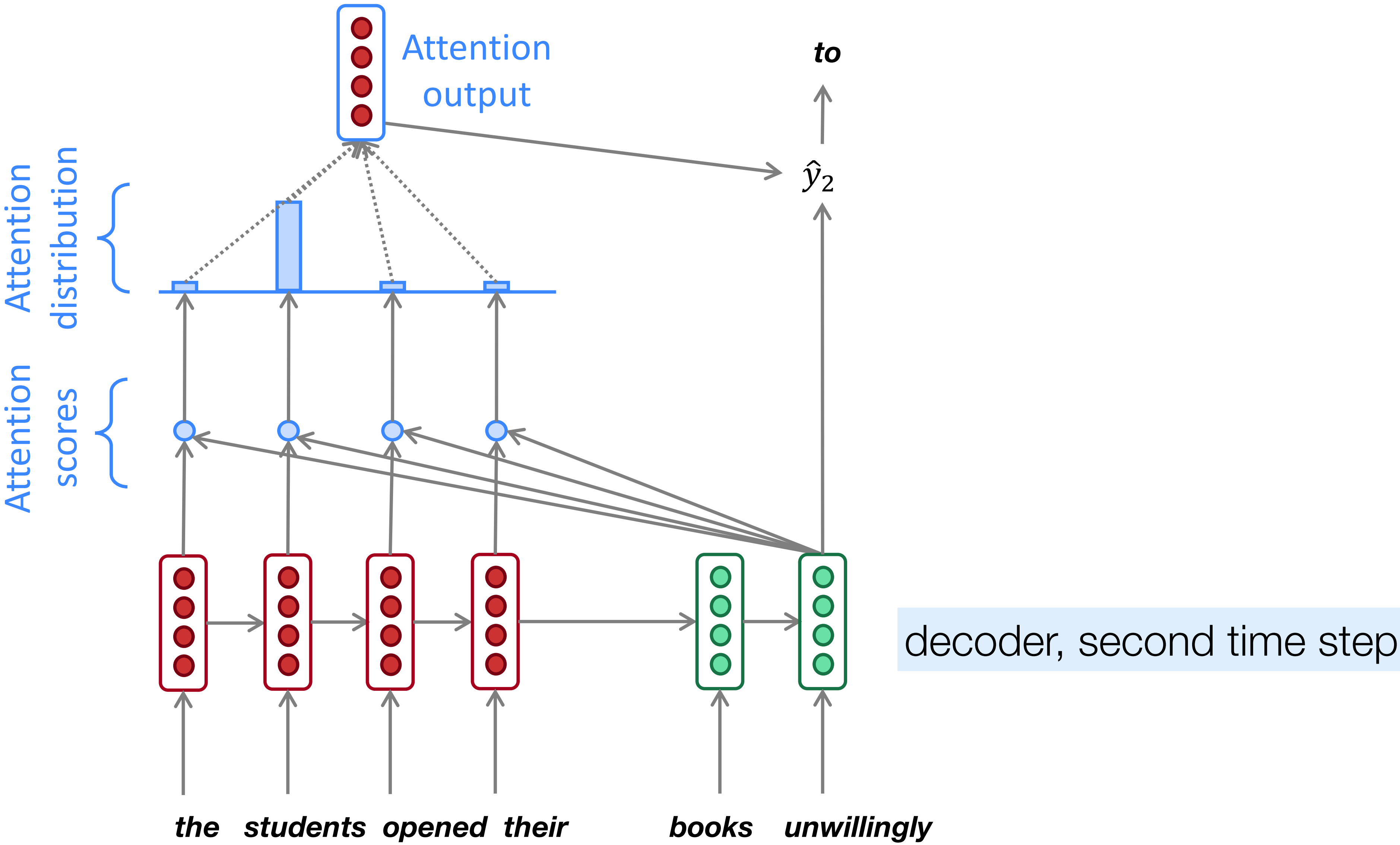
Sequence-to-sequence with attention



Concatenate (or otherwise compose) the attention output with the current hidden state, then pass through a softmax layer to predict the next word

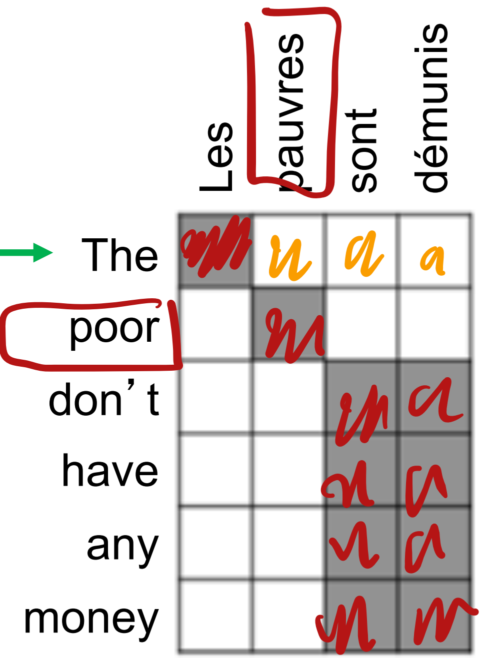
Slides adapted from Mohit Iyyer

Sequence-to-sequence with attention



Slides adapted from Mohit Iyyer

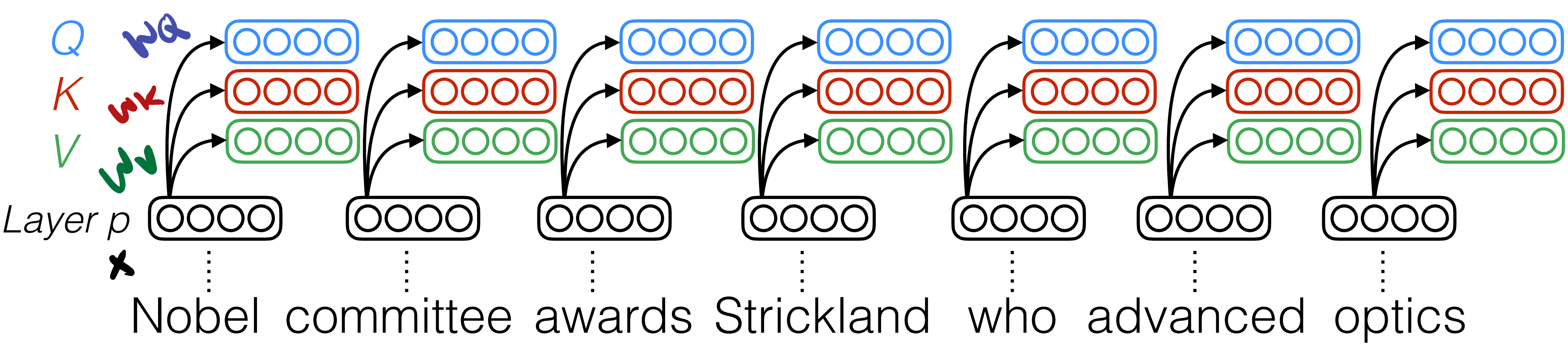
- Attention solves the bottleneck problem
  - Attention allows decoder to look directly at source; bypass bottleneck
- Attention helps with vanishing gradient problem
  - Provides shortcut to faraway states
- Attention provides some interpretability
  - By inspecting attention distribution, we can see what the decoder was focusing on
  - We get alignment for free!
  - This is cool because we never explicitly trained an alignment system
  - The network just learned alignment by itself



Slides by Emma Strubell!

[Vaswani et al. 2017]

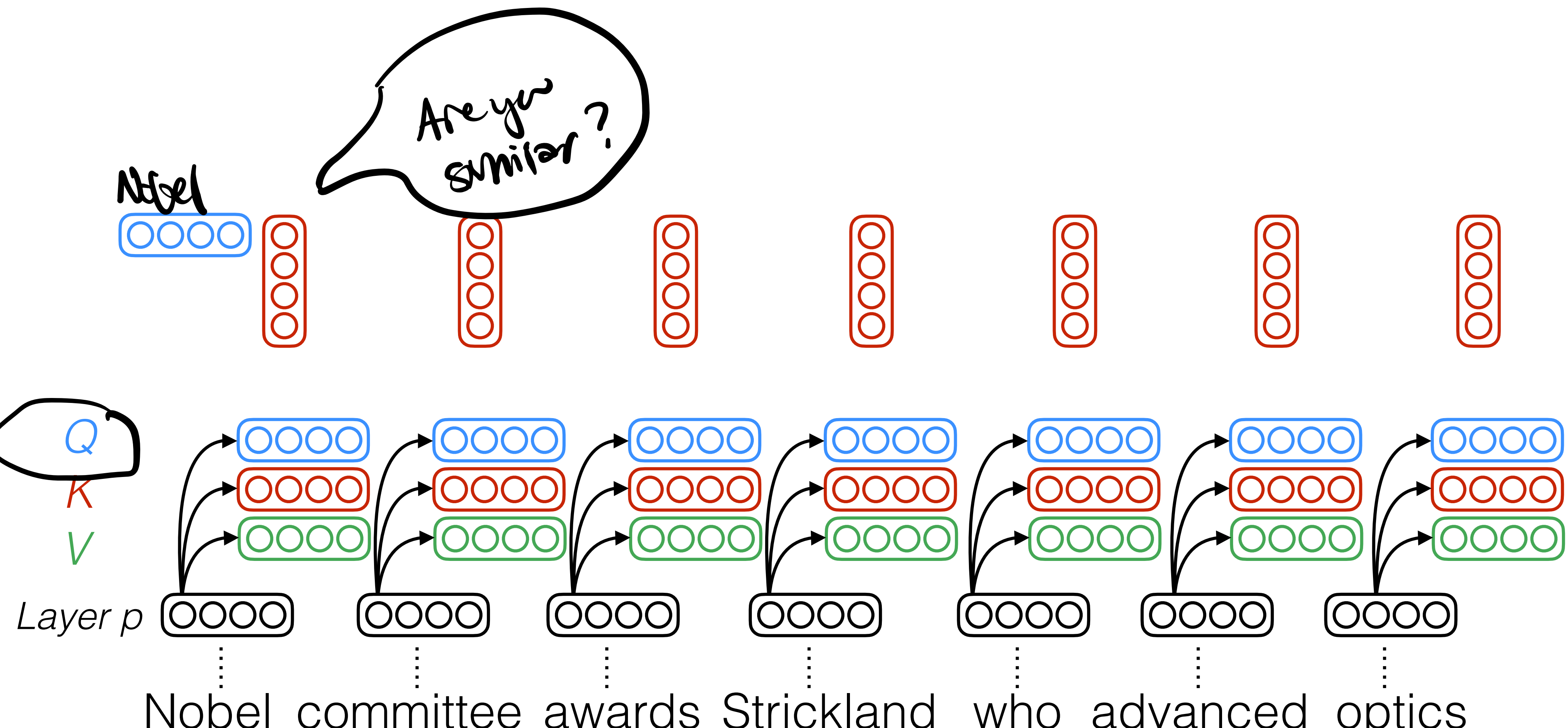
Self-attention



Slides by Emma Strubell!

[Vaswani et al. 2017]

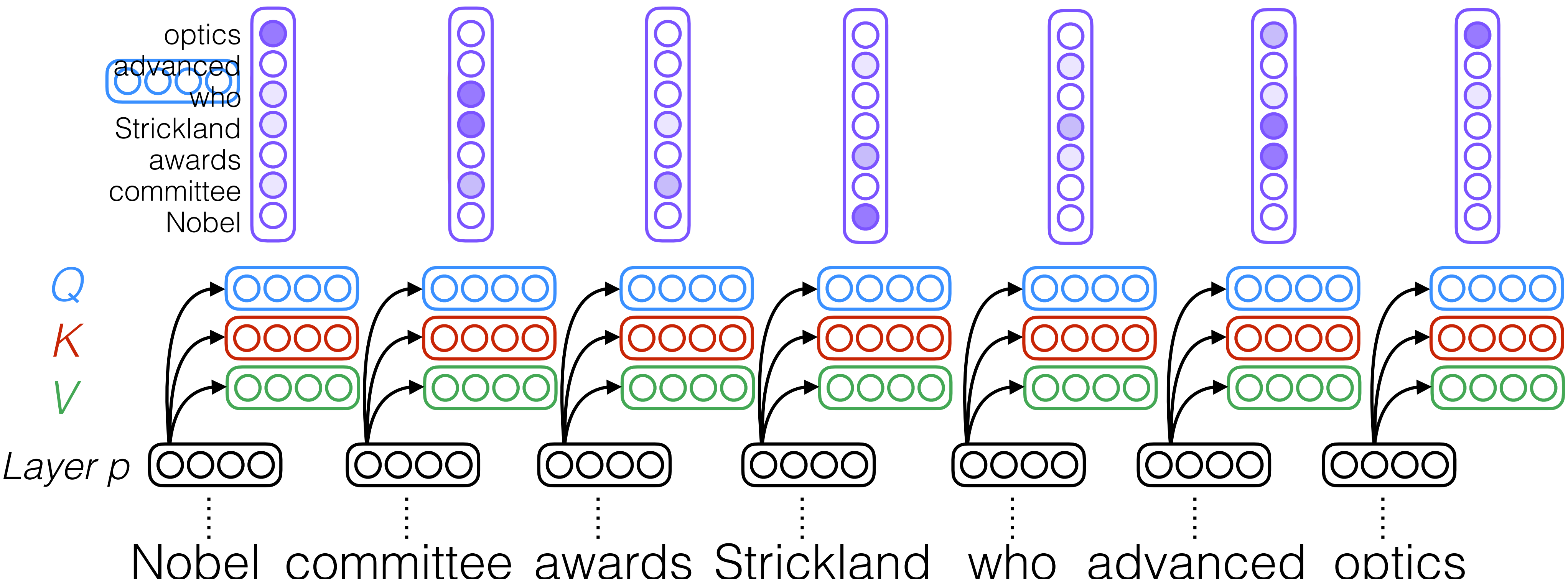
Self-attention



Slides by Emma Strubell!

[Vaswani et al. 2017]

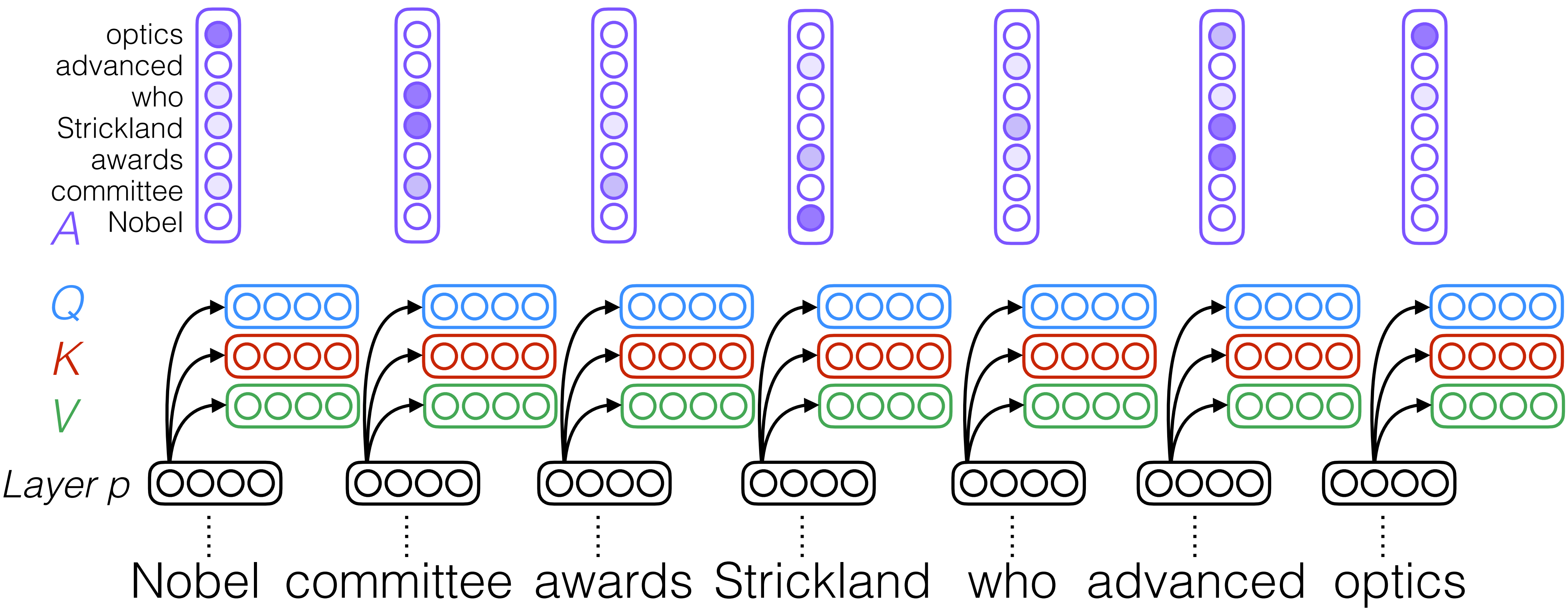
Self-attention



Slides by Emma Strubell!

[Vaswani et al. 2017]

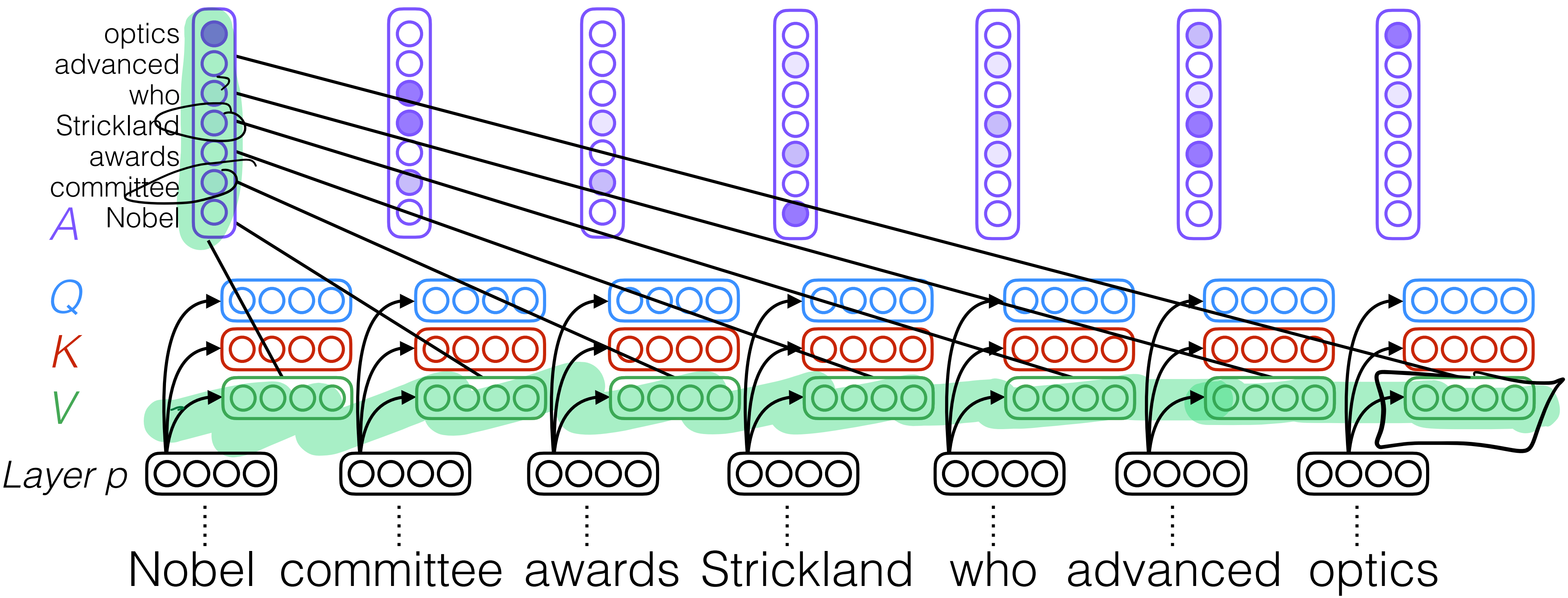
Self-attention



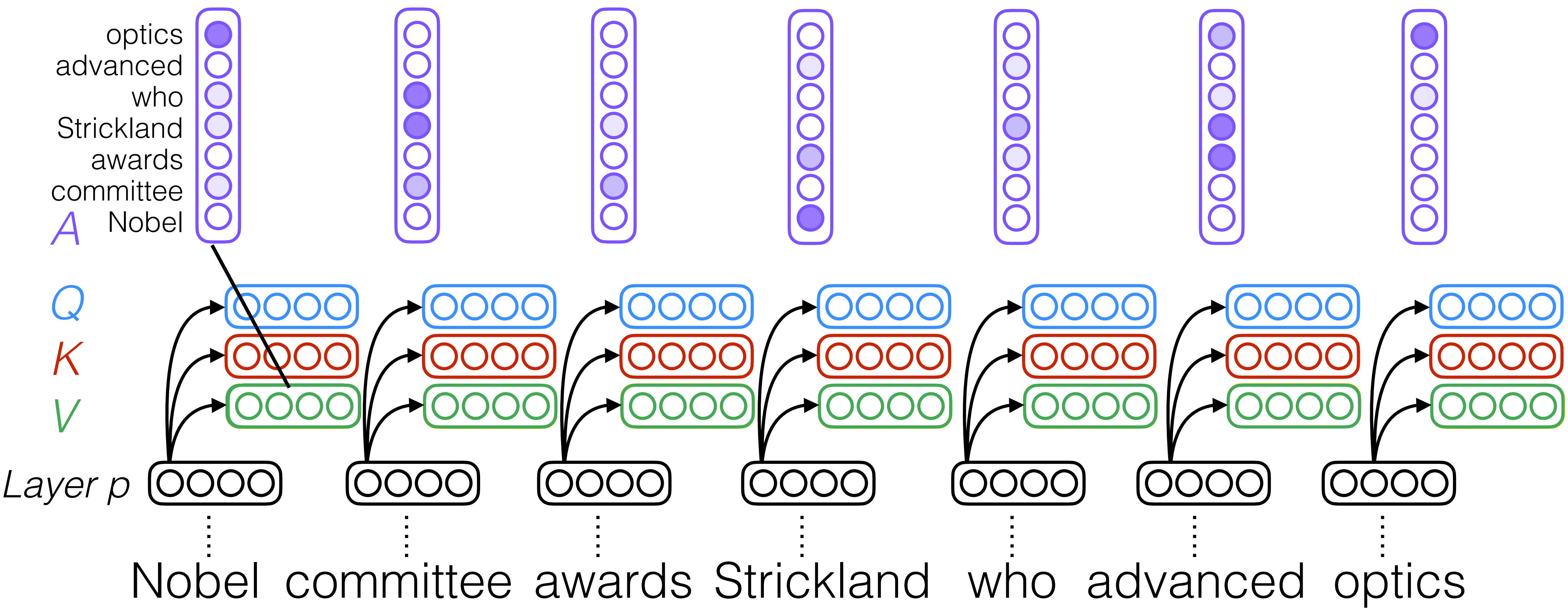
Slides by Emma Strubell!

[Vaswani et al. 2017]

Self-attention



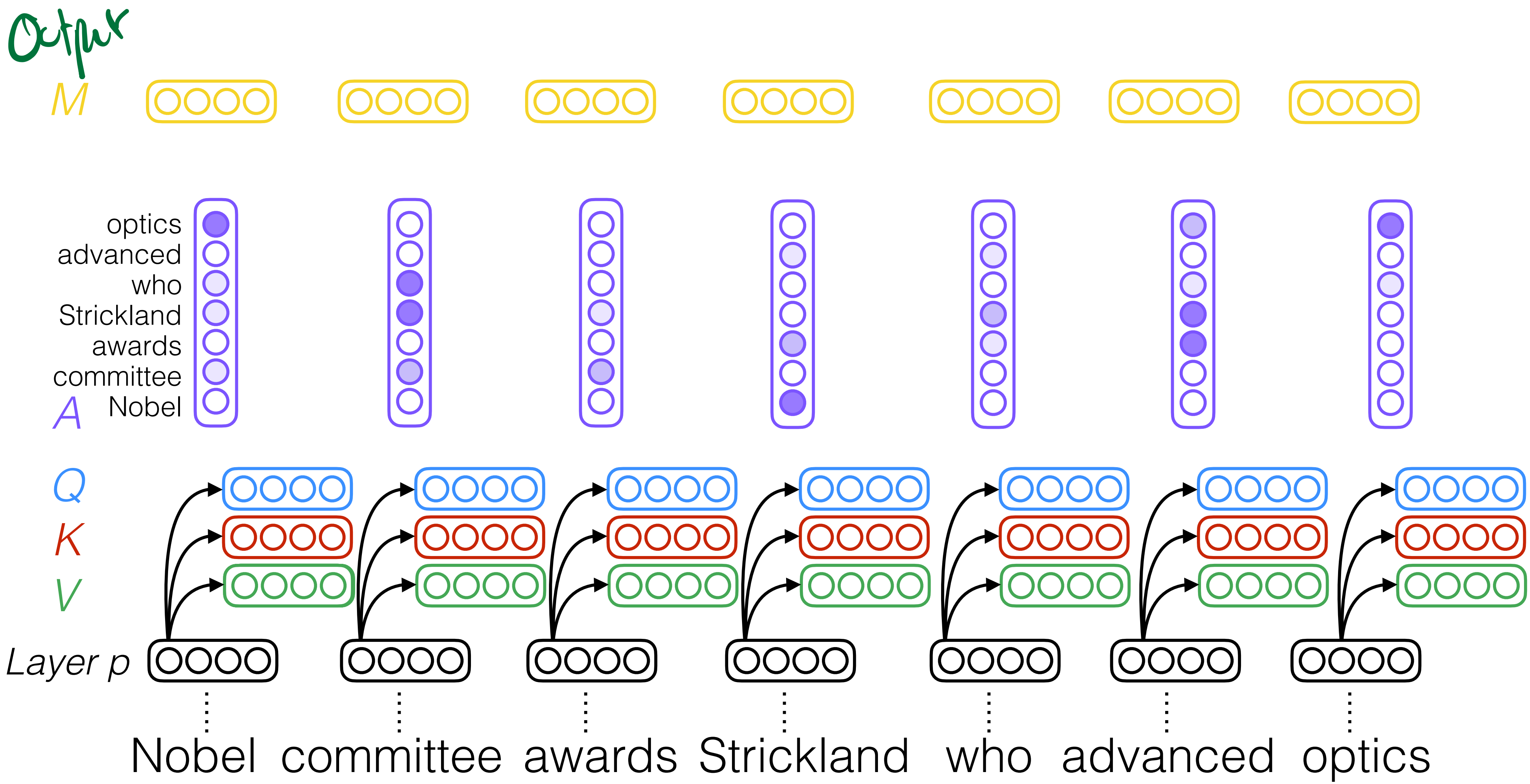
Self-attention



Slides by Emma Strubell!

[Vaswani et al. 2017]

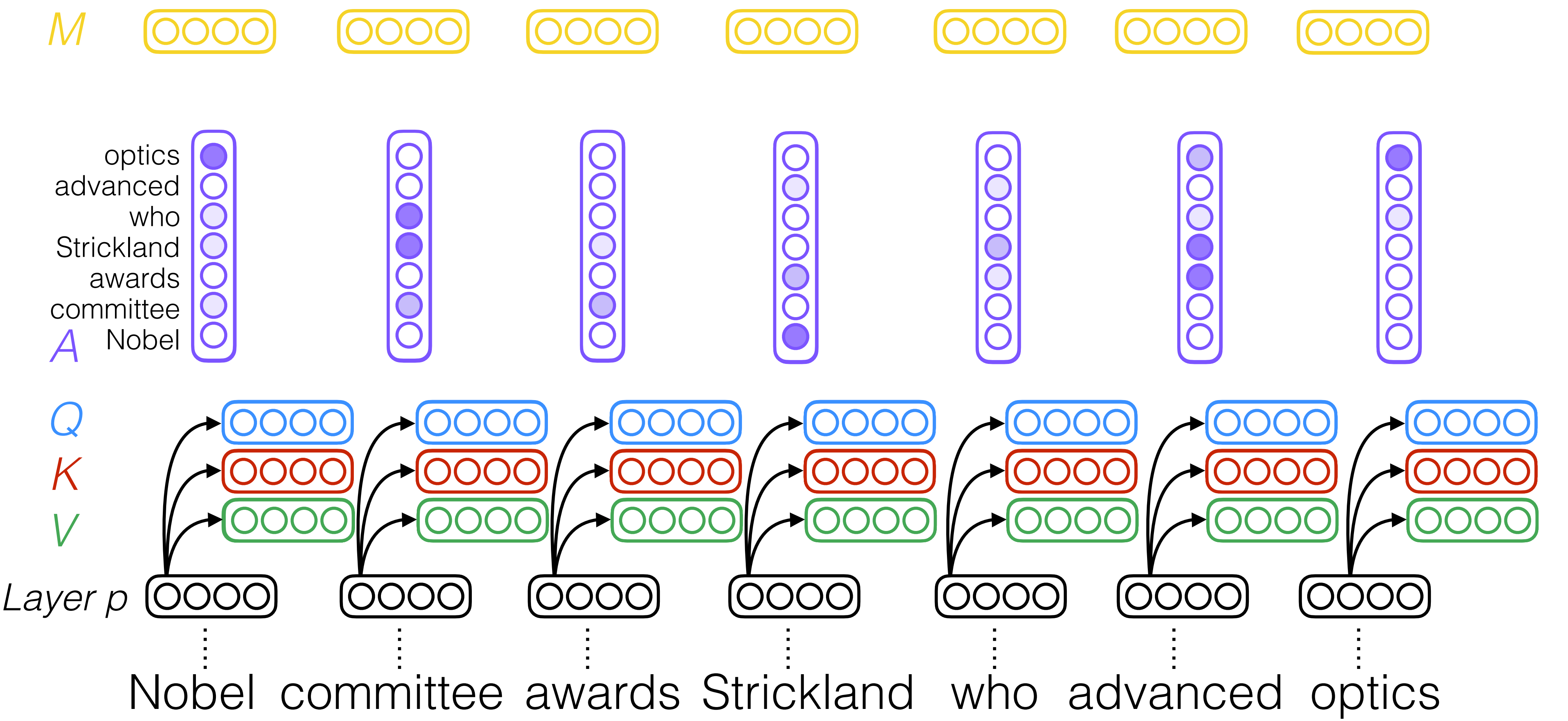
Self-attention



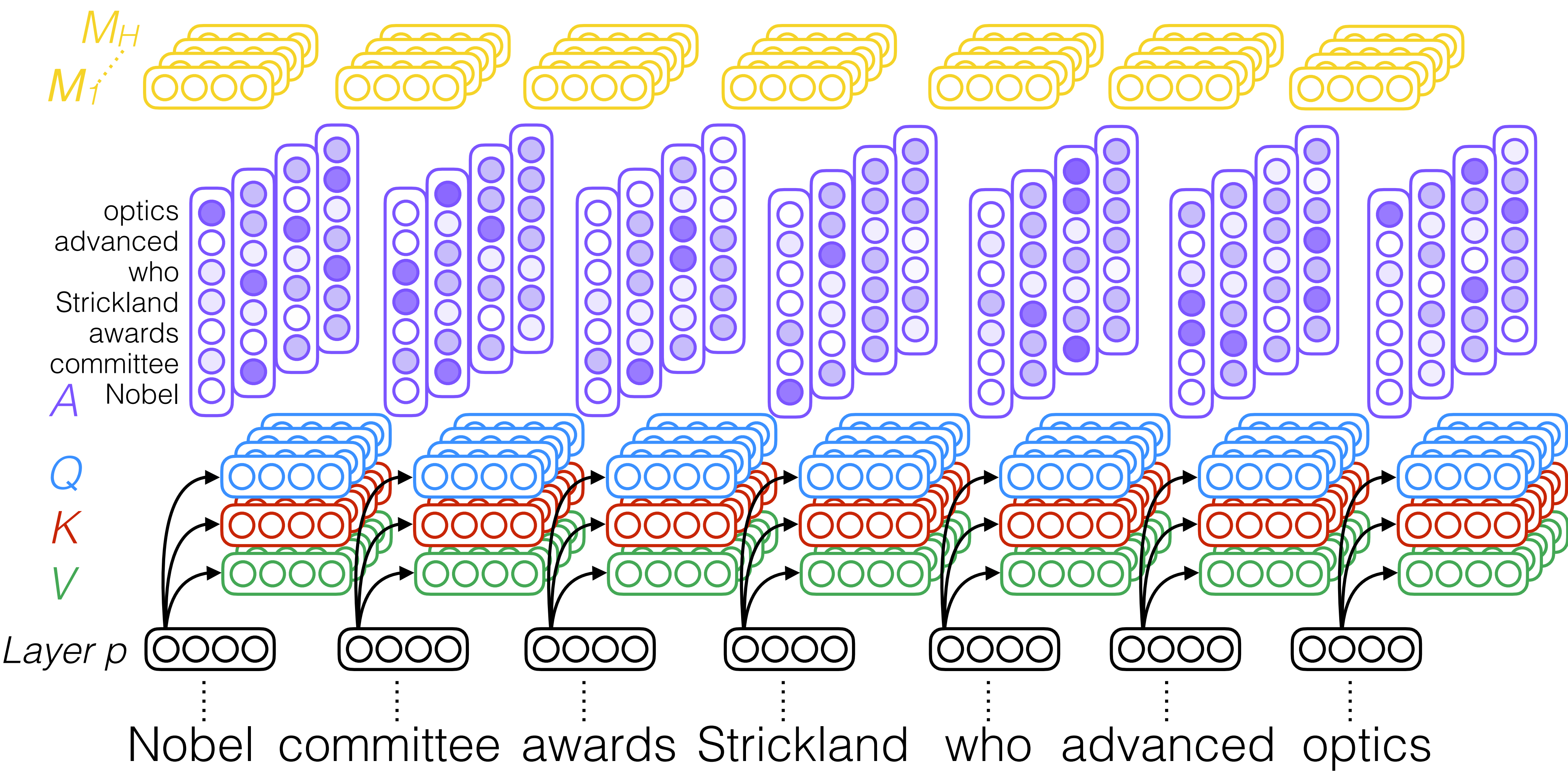
Slides by Emma Strubell!

[Vaswani et al. 2017]

Self-attention

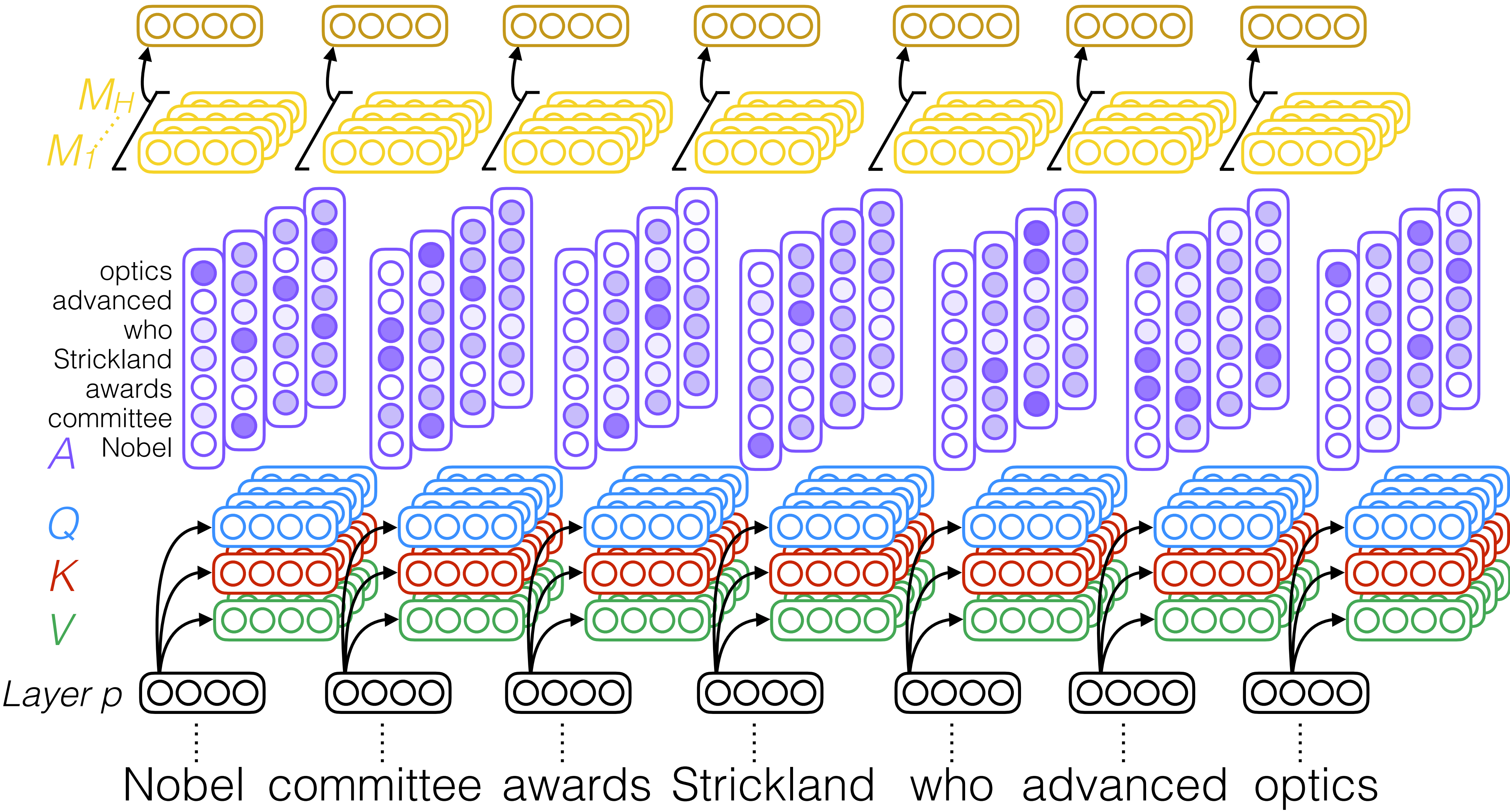


Multi-head self-attention



[Vaswani et al. 2017]

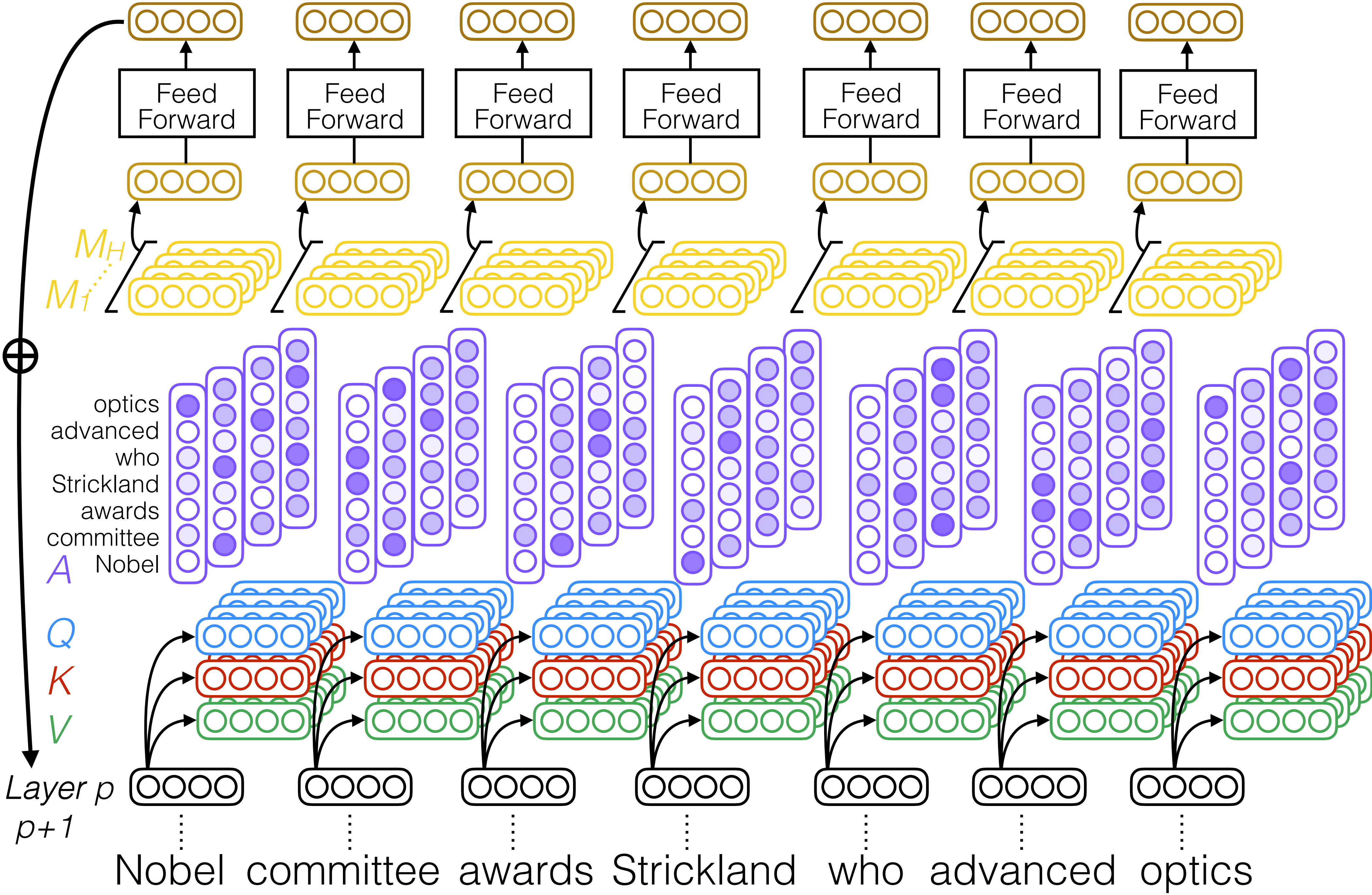
Multi-head self-attention



Slides by Emma Strubell!

[Vaswani et al. 2017]

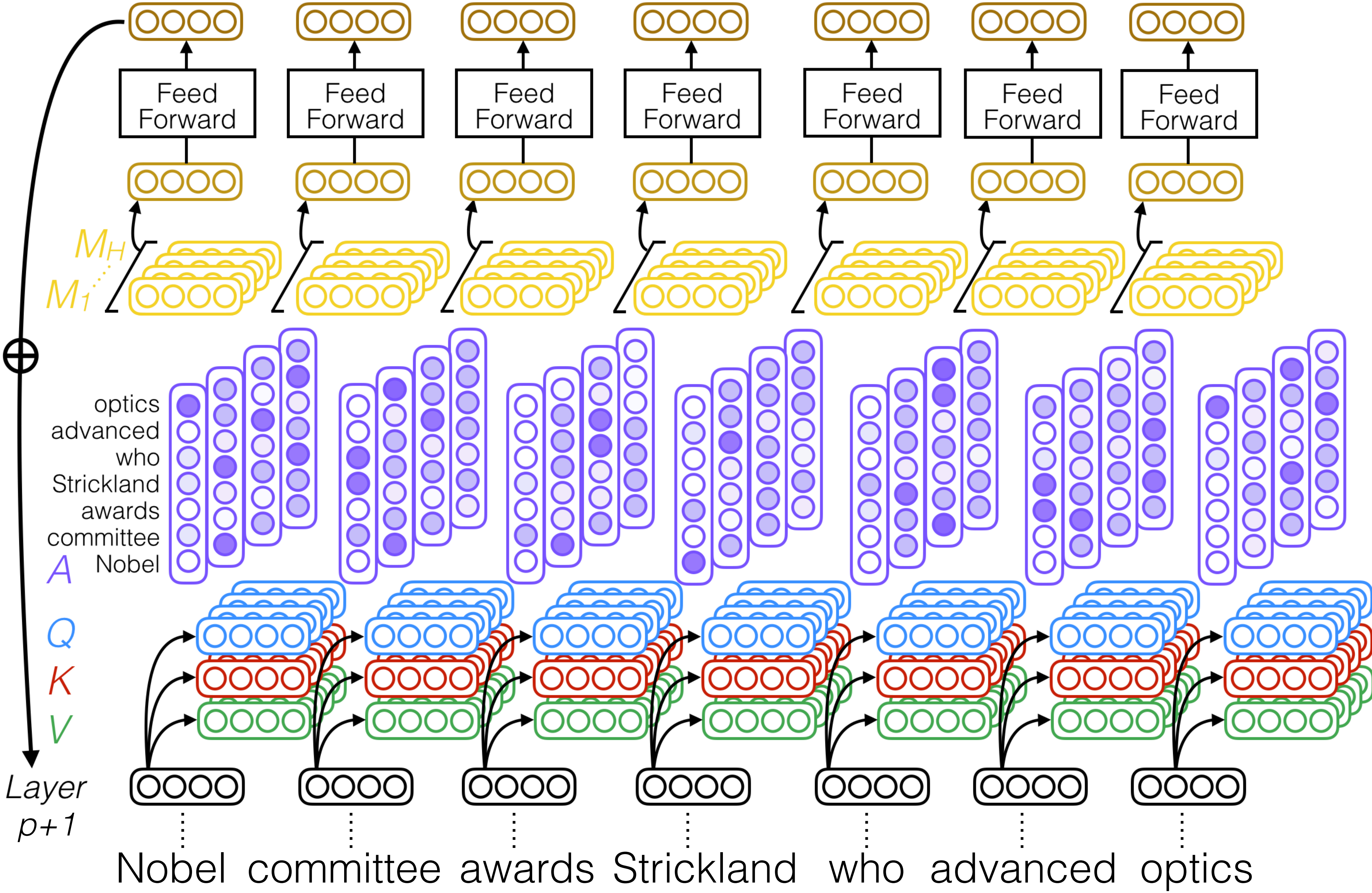
Multi-head self-attention



Slides by Emma Strubell!

[Vaswani et al. 2017]

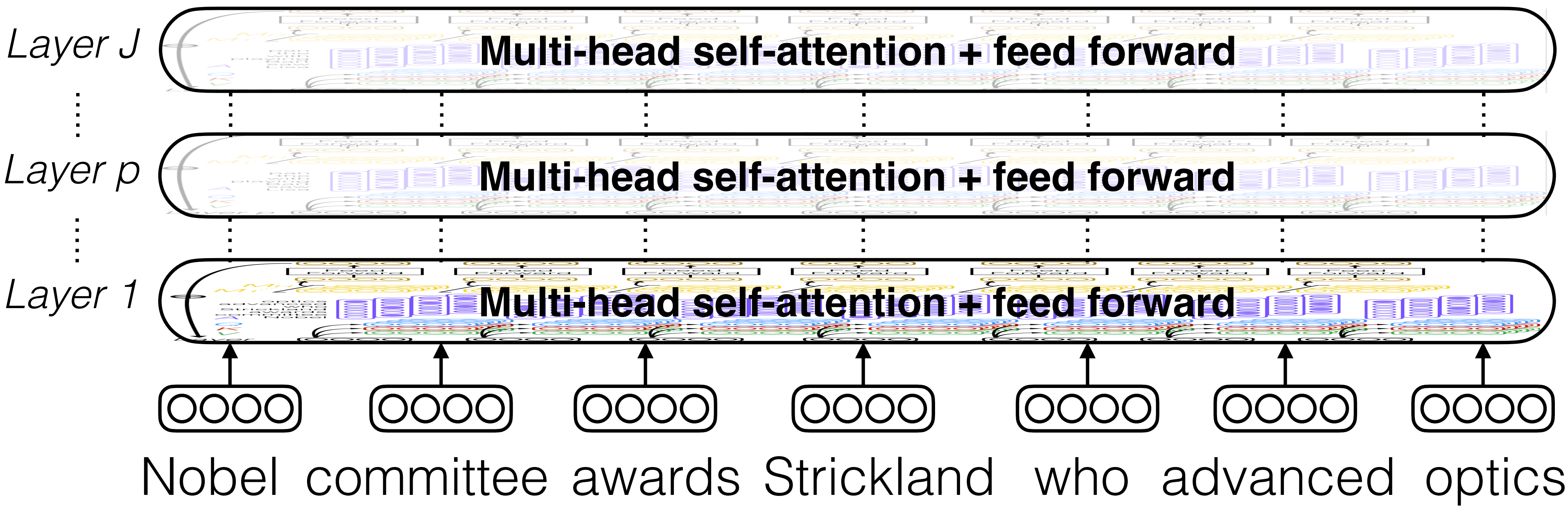
Multi-head self-attention



Slides by Emma Strubell!

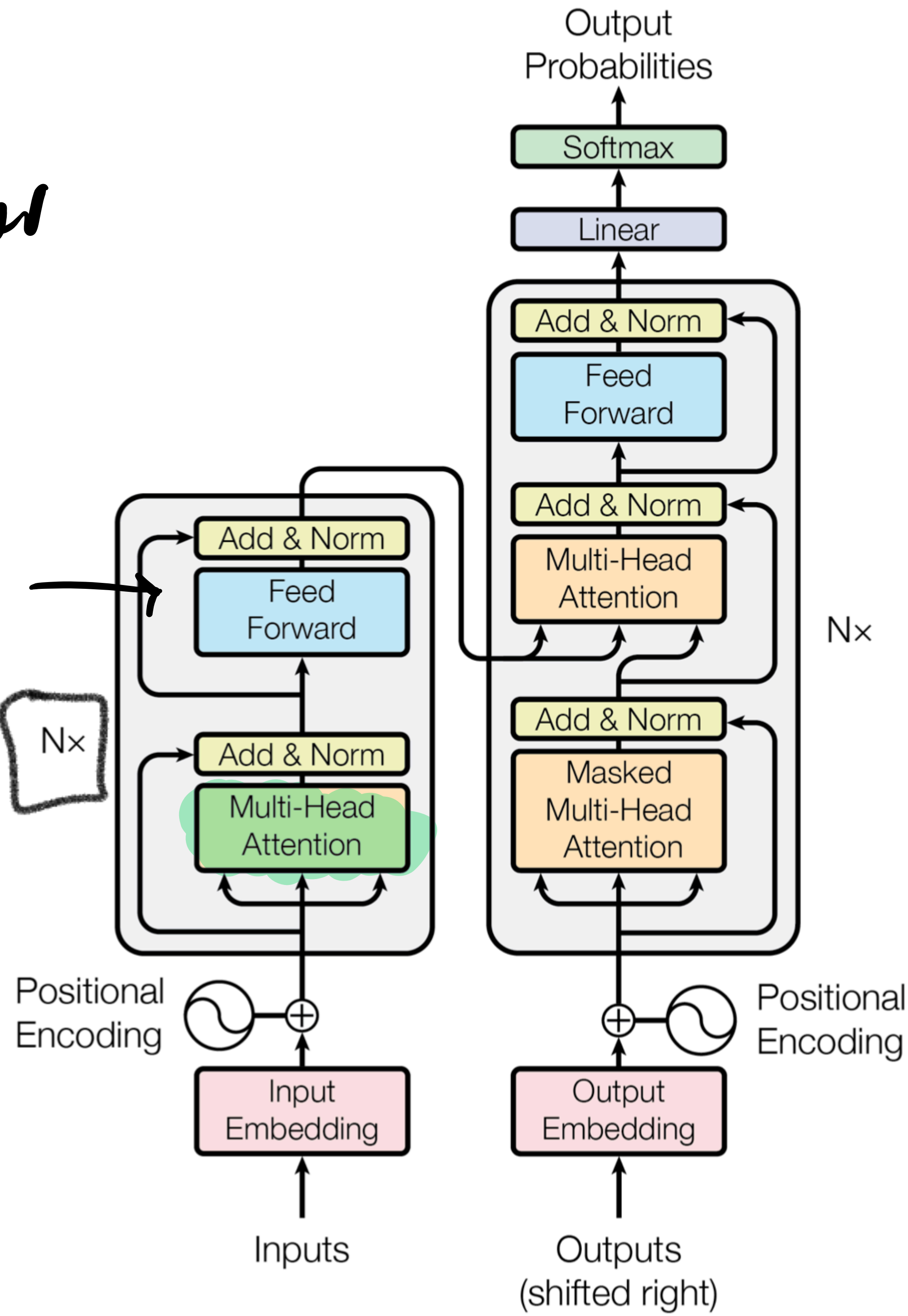
[Vaswani et al. 2017]

Multi-head self-attention

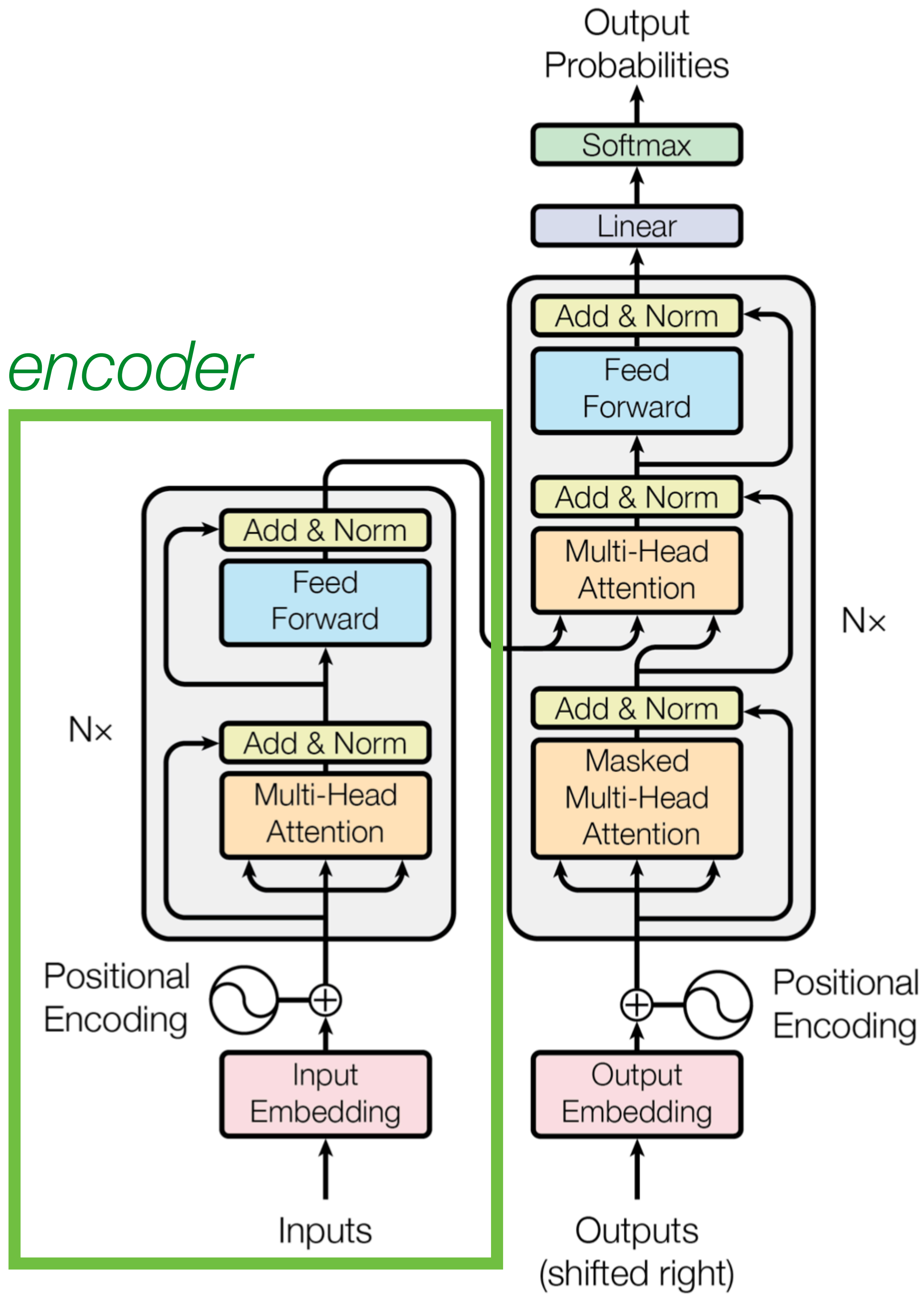


Transformers

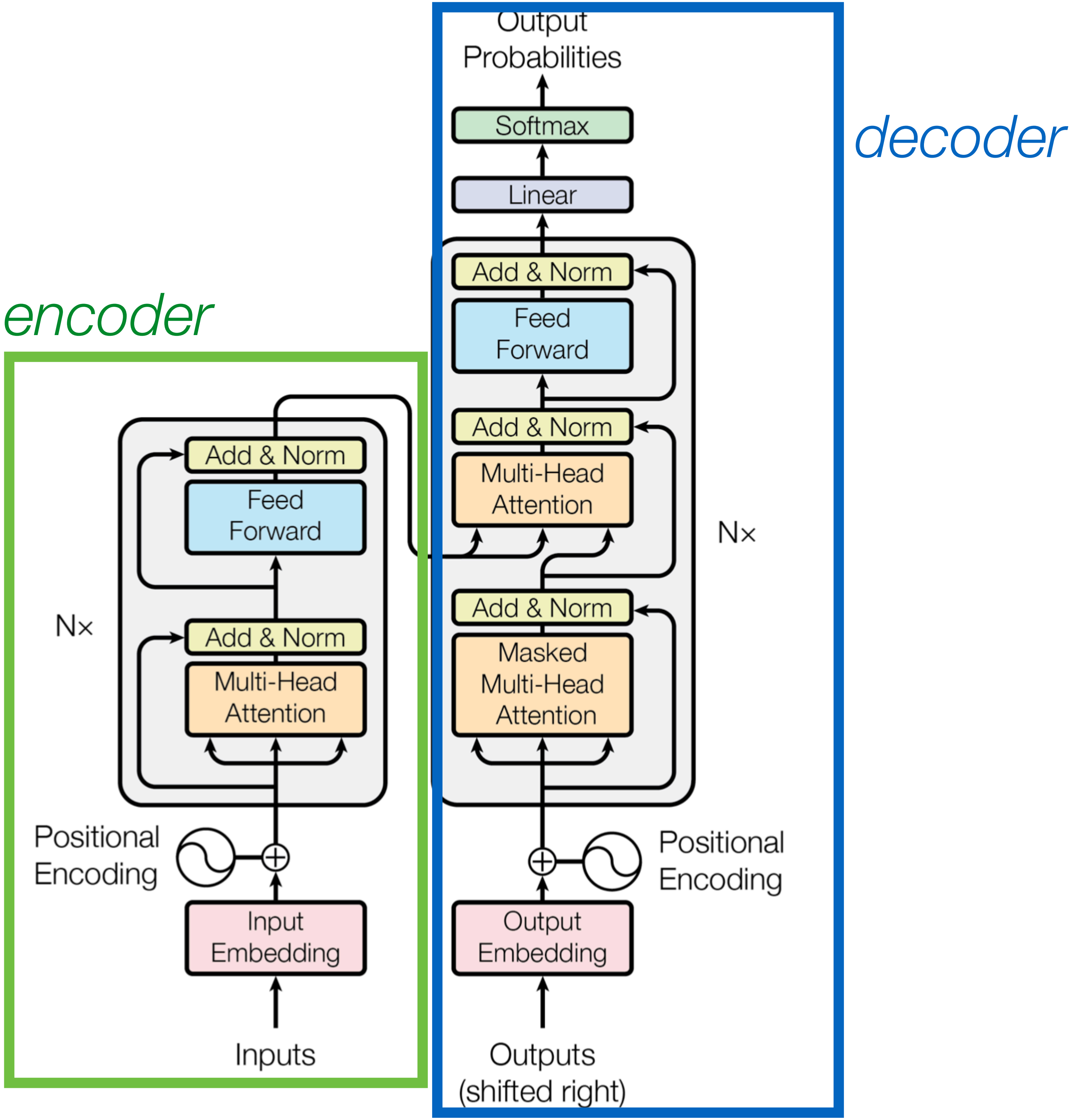
Transformer



Slides adapted from Mohit Iyyer

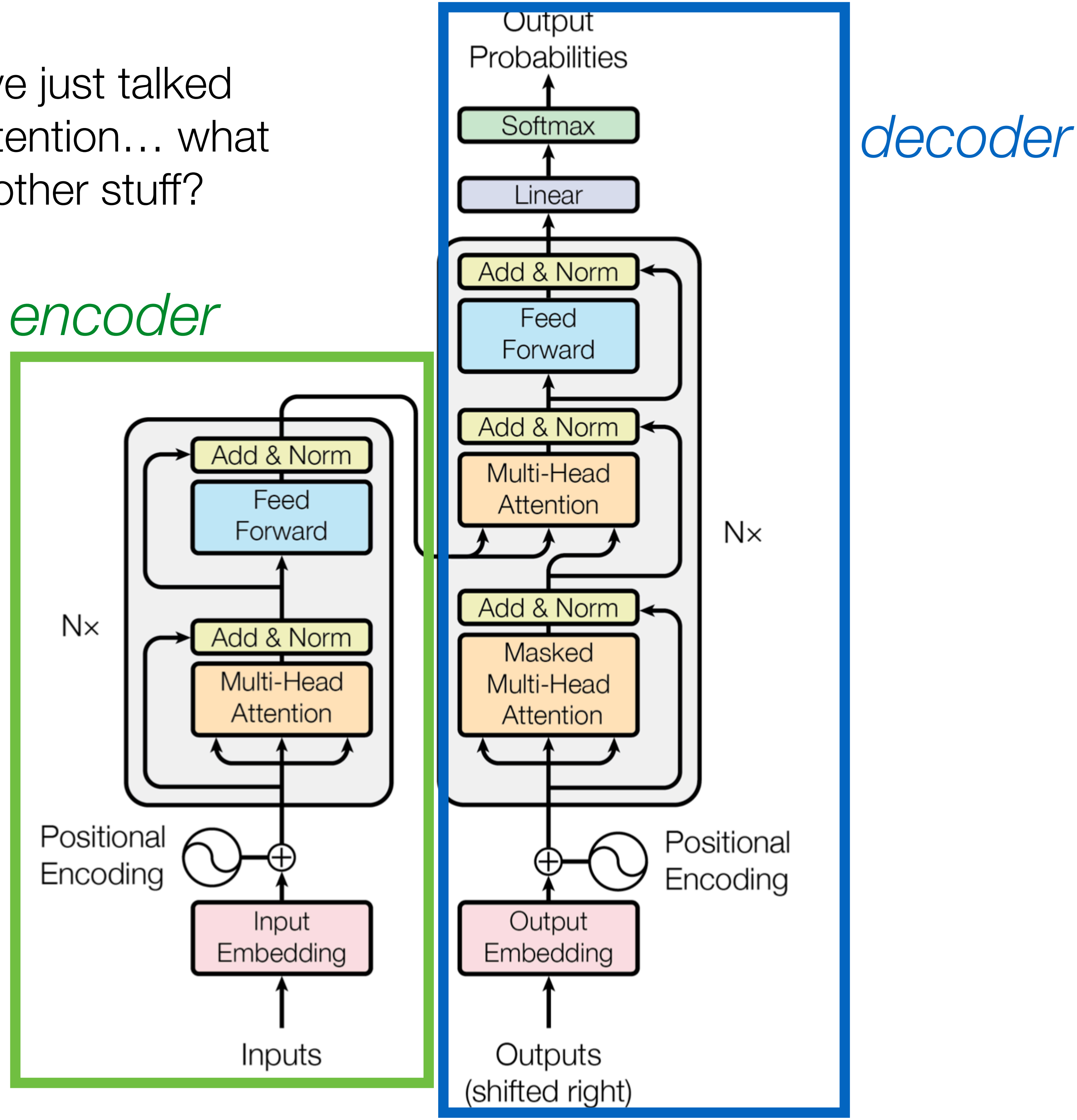


Slides adapted from Mohit Iyyer

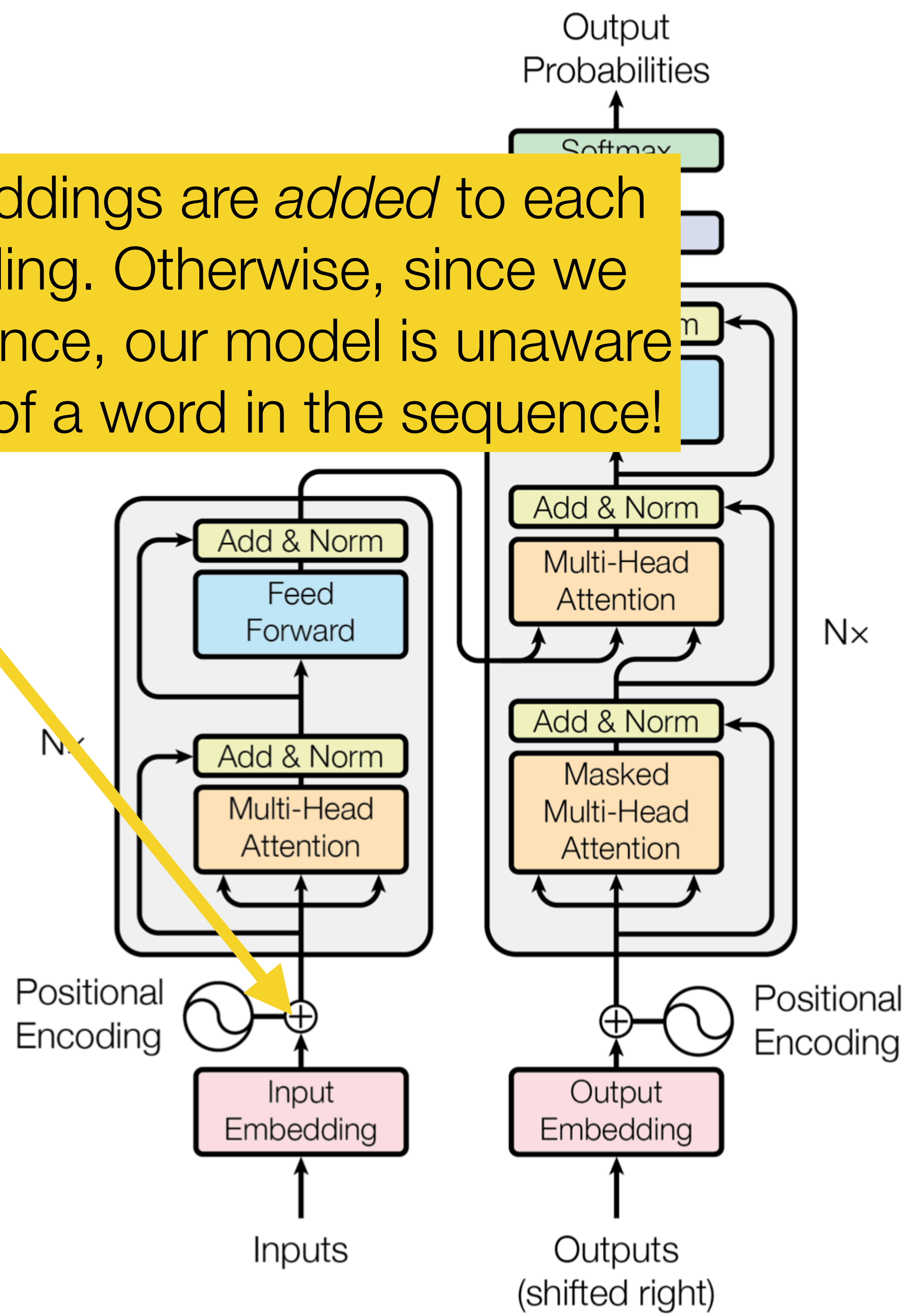


encoder
decoder
Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted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

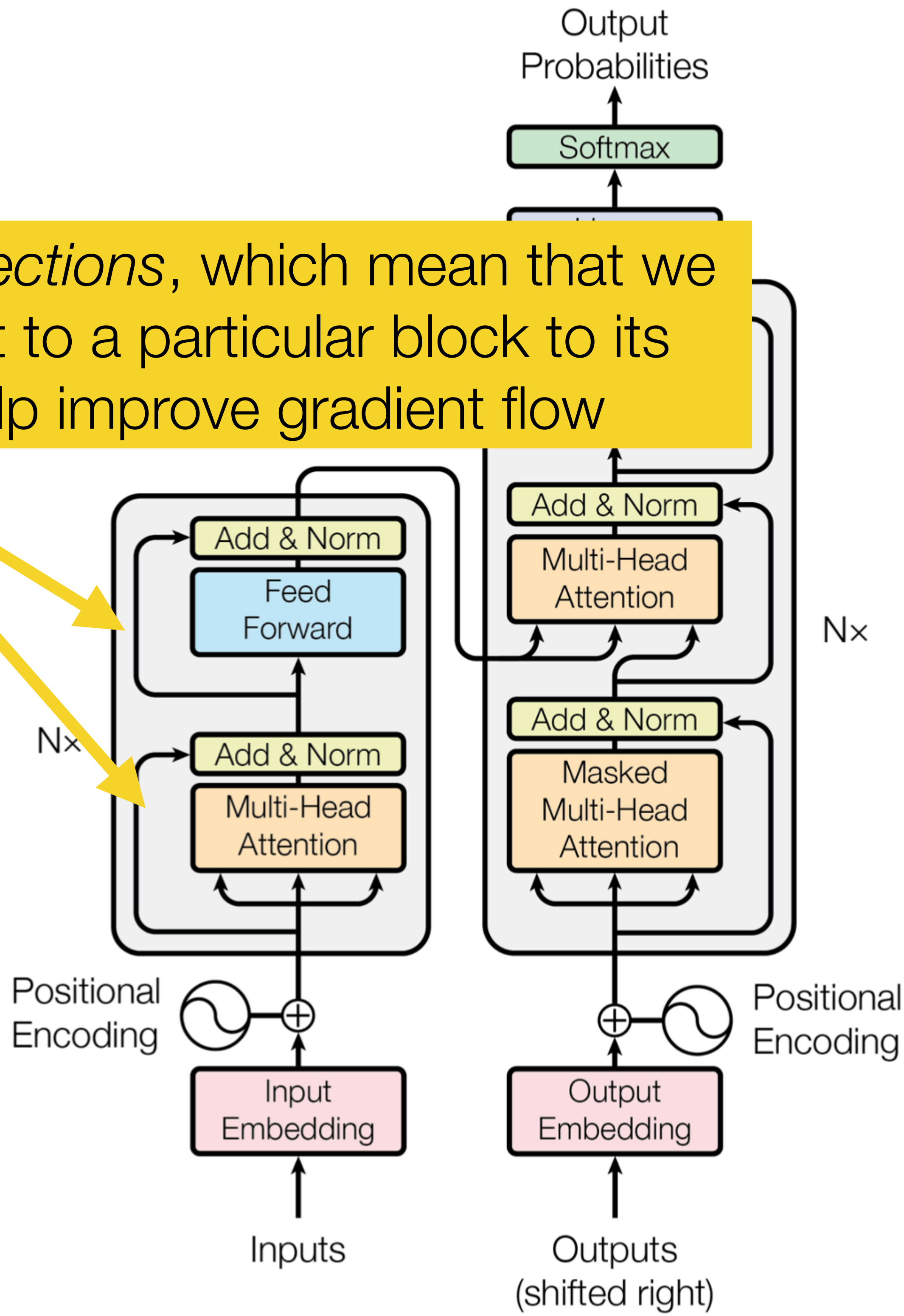
So far we've just talked about self-attention… what is all this other stuff?



Position embeddings are *added* to each word embedding. Otherwise, since we have no recurrence, our model is unaware of the position of a word in the sequence!

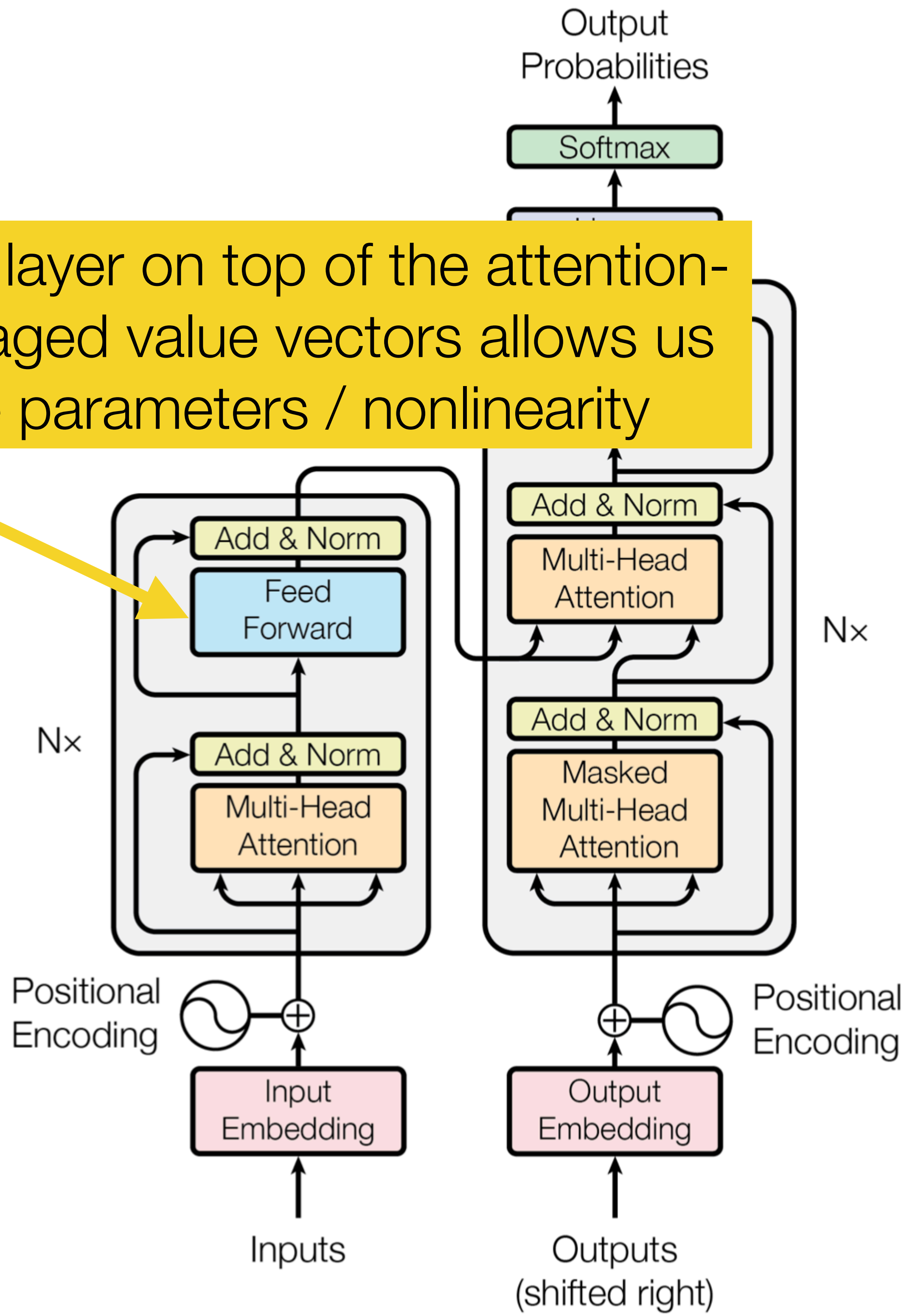


Residual connections, which mean that we add the input to a particular block to its output, help improve gradient flow



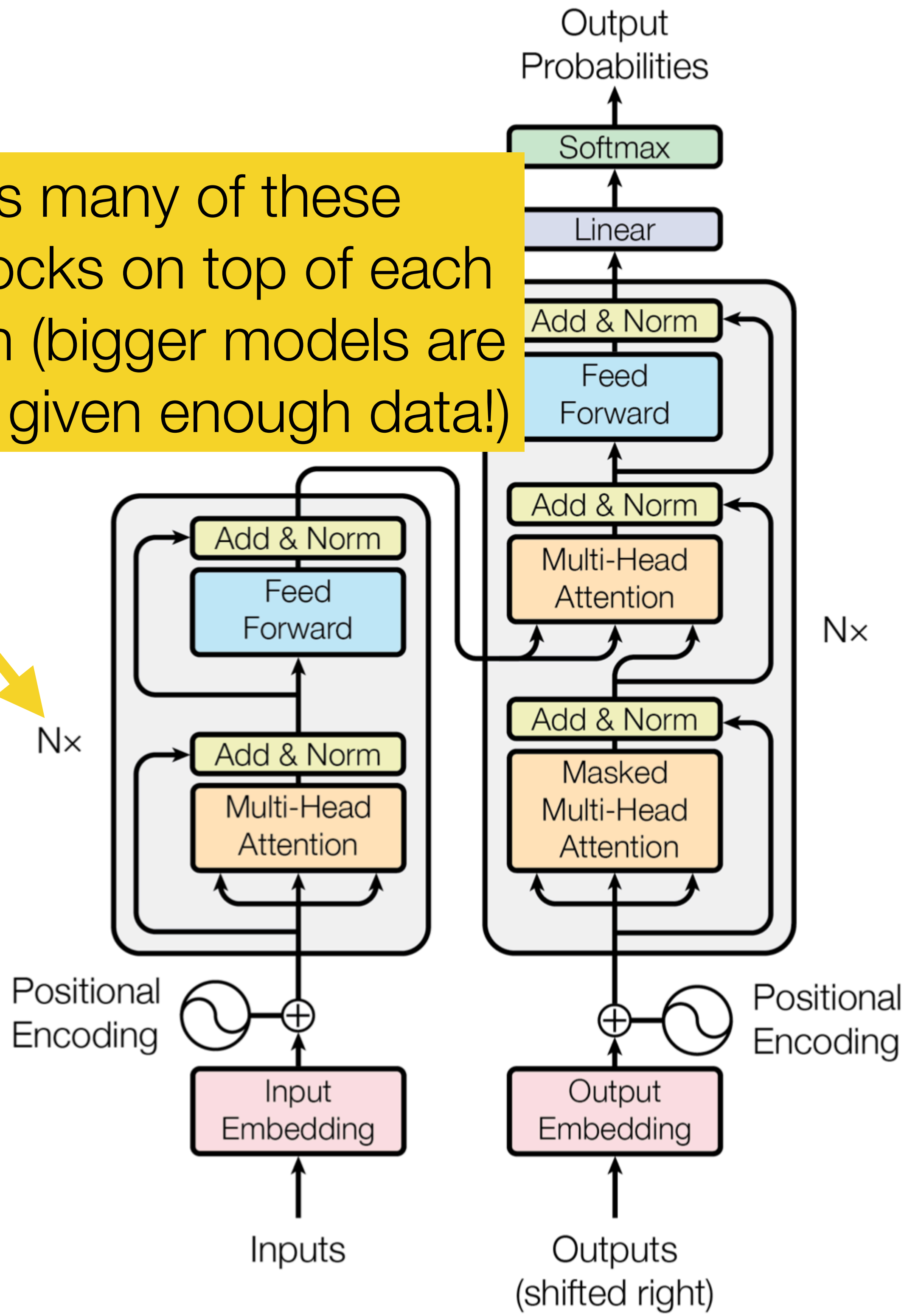
Slides adapted from Mohit Iyyer

A feed-forward layer on top of the attention-weighted averaged value vectors allows us to add more parameters / nonlinearity



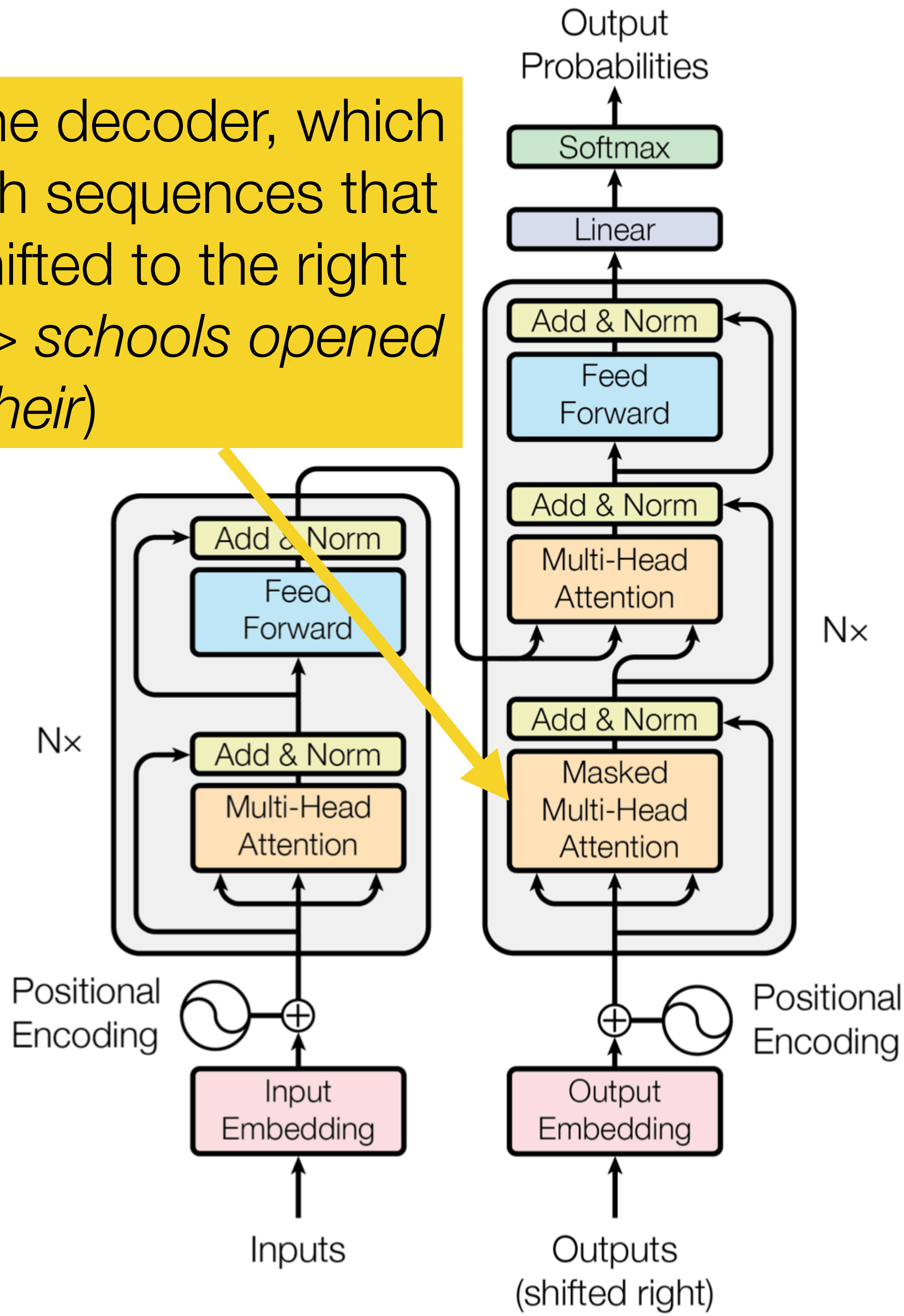
Slides adapted from Mohit Iyyer

We stack as many of these *Transformer* blocks on top of each other as we can (bigger models are generally better given enough data!)

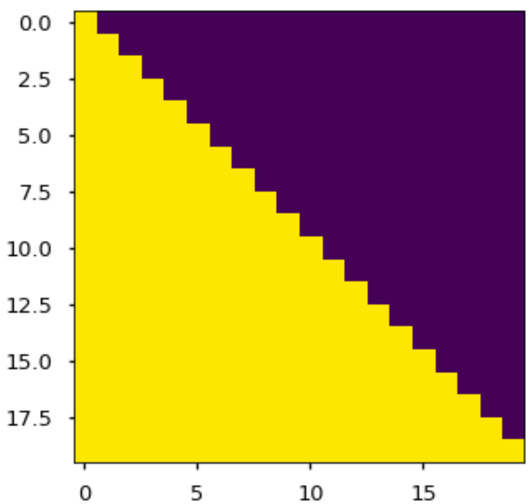


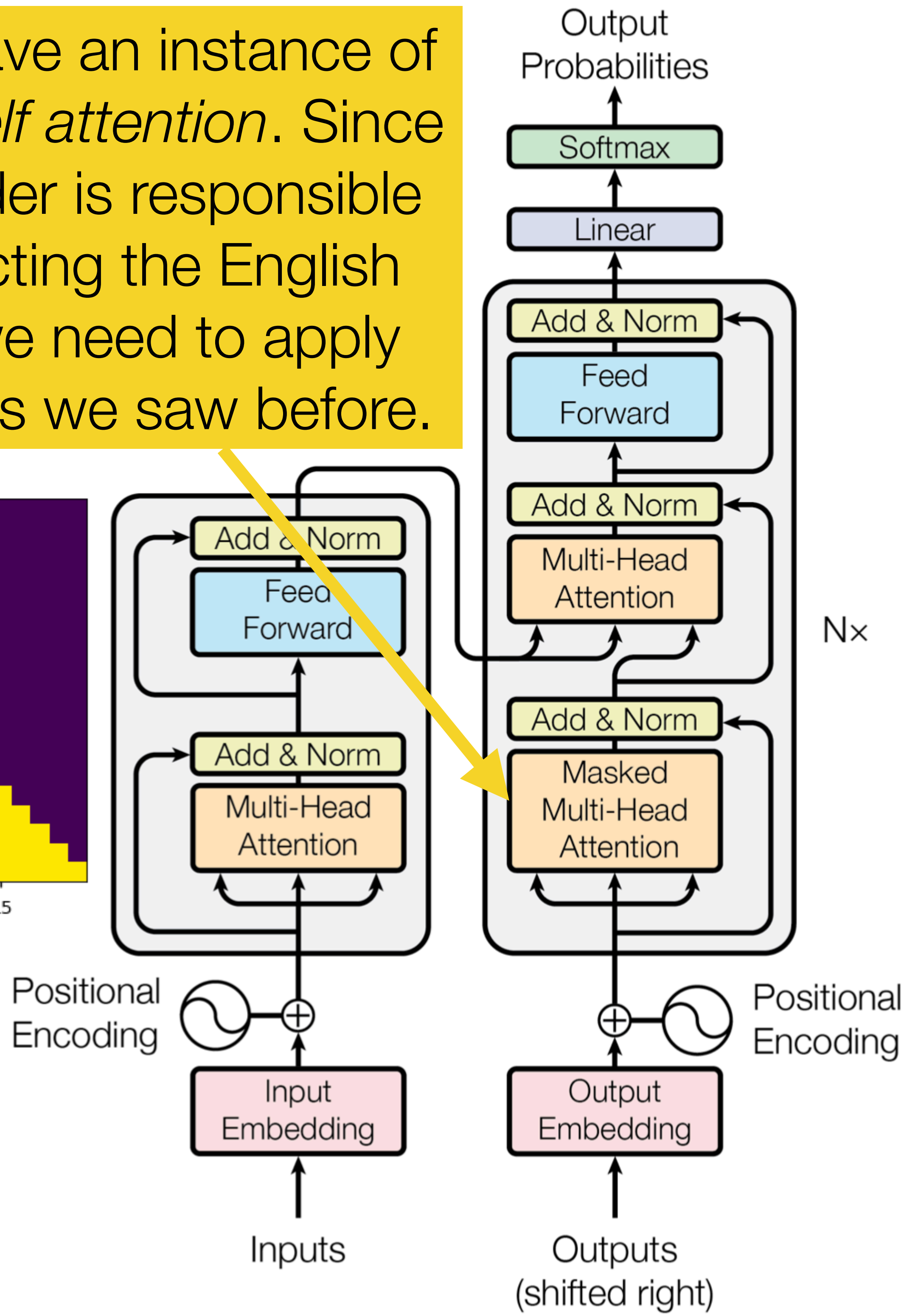
Slides adapted from Mohit Iyyer

Moving onto the decoder, which takes in English sequences that have been shifted to the right (e.g., *<START> schools opened their*)

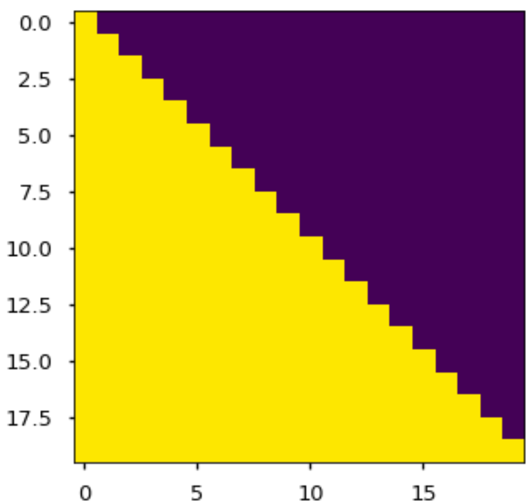


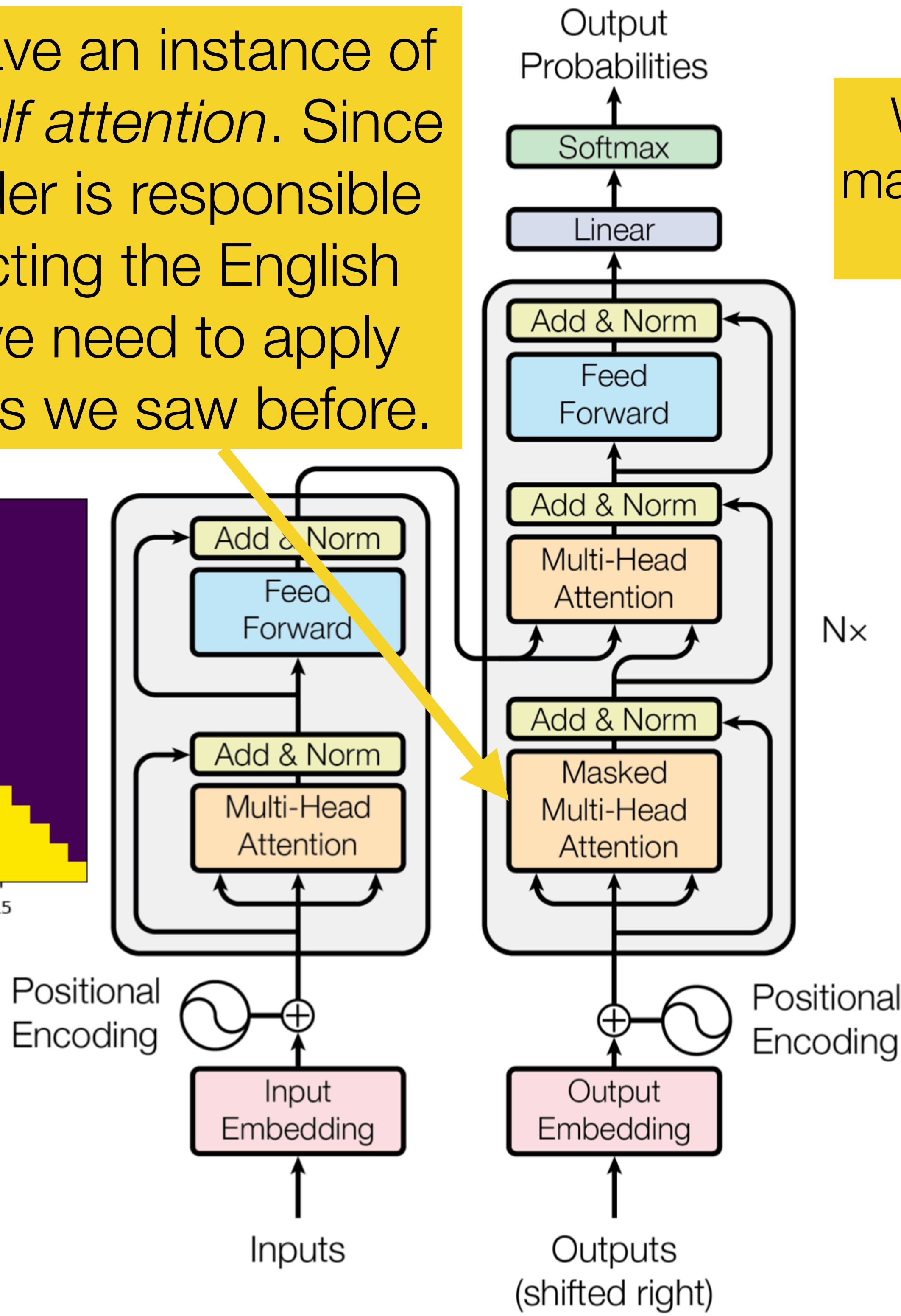
We first have an instance of *masked self attention*. Since the decoder is responsible for predicting the English words, we need to apply masking as we saw before.





We first have an instance of *masked self attention*. Since the decoder is responsible for predicting the English words, we need to apply masking as we saw before.

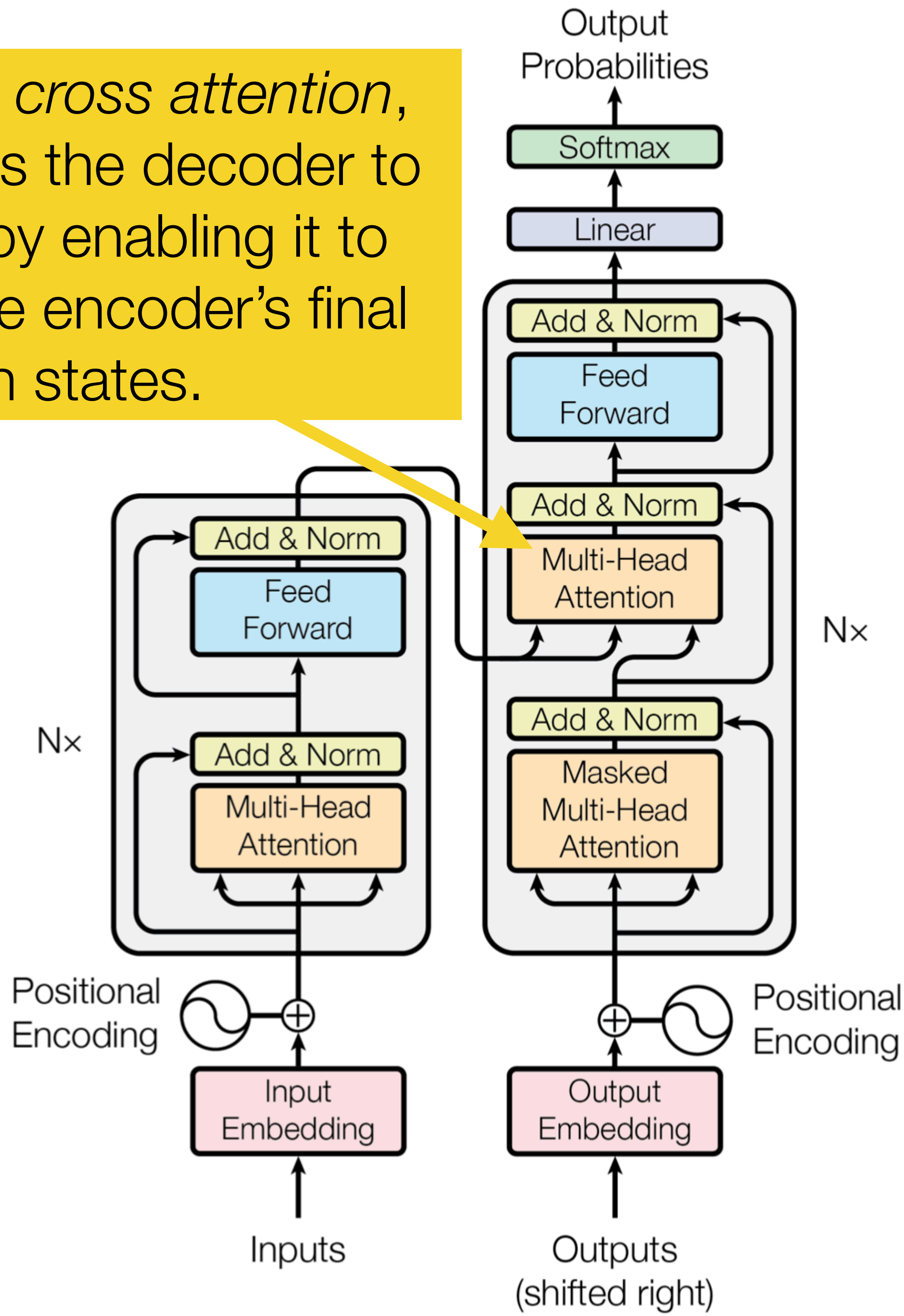




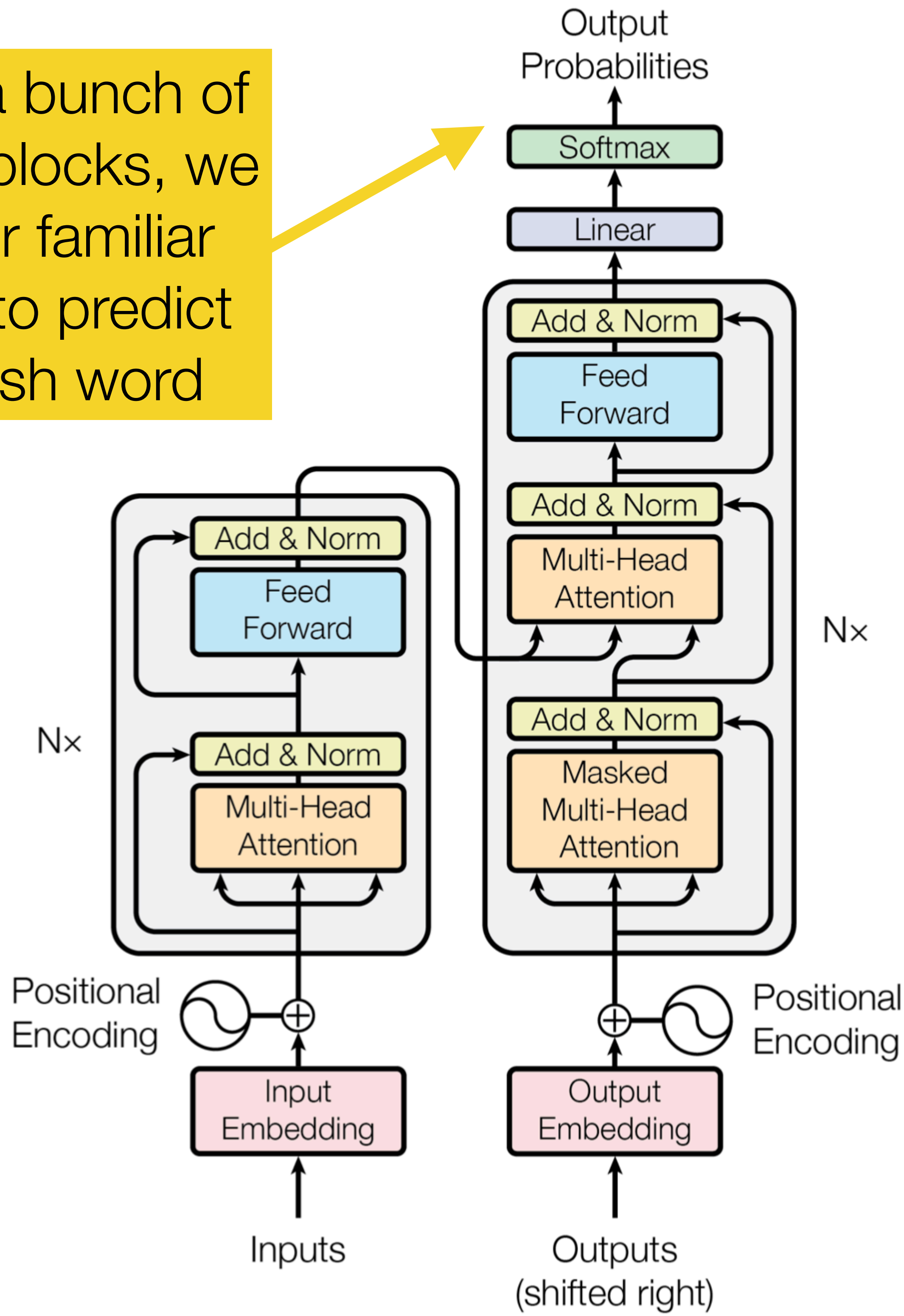
Why don't we do masked self-attention in the encoder?

Slides adapted from Mohit Iyyer

Now, we have *cross attention*, which connects the decoder to the encoder by enabling it to attend over the encoder's final hidden states.



After stacking a bunch of these decoder blocks, we finally have our familiar Softmax layer to predict the next English word



Slides adapted from Mohit Iyyer