# CS 333: Natural Language Processing

Fall 2023

Prof. Carolyn Anderson
Wellesley College

# Reminders

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

# Recap

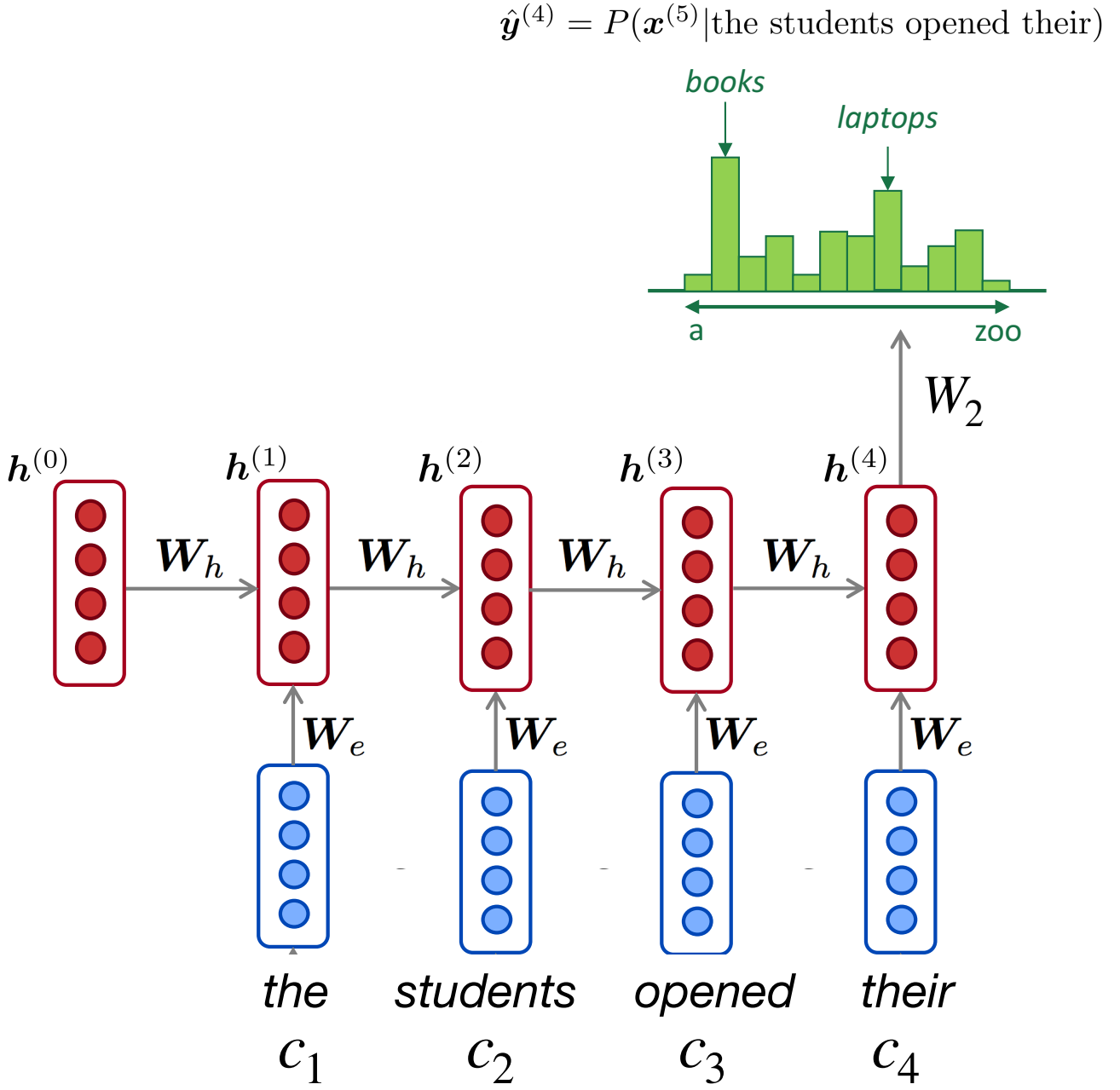
# why is this good?

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**:

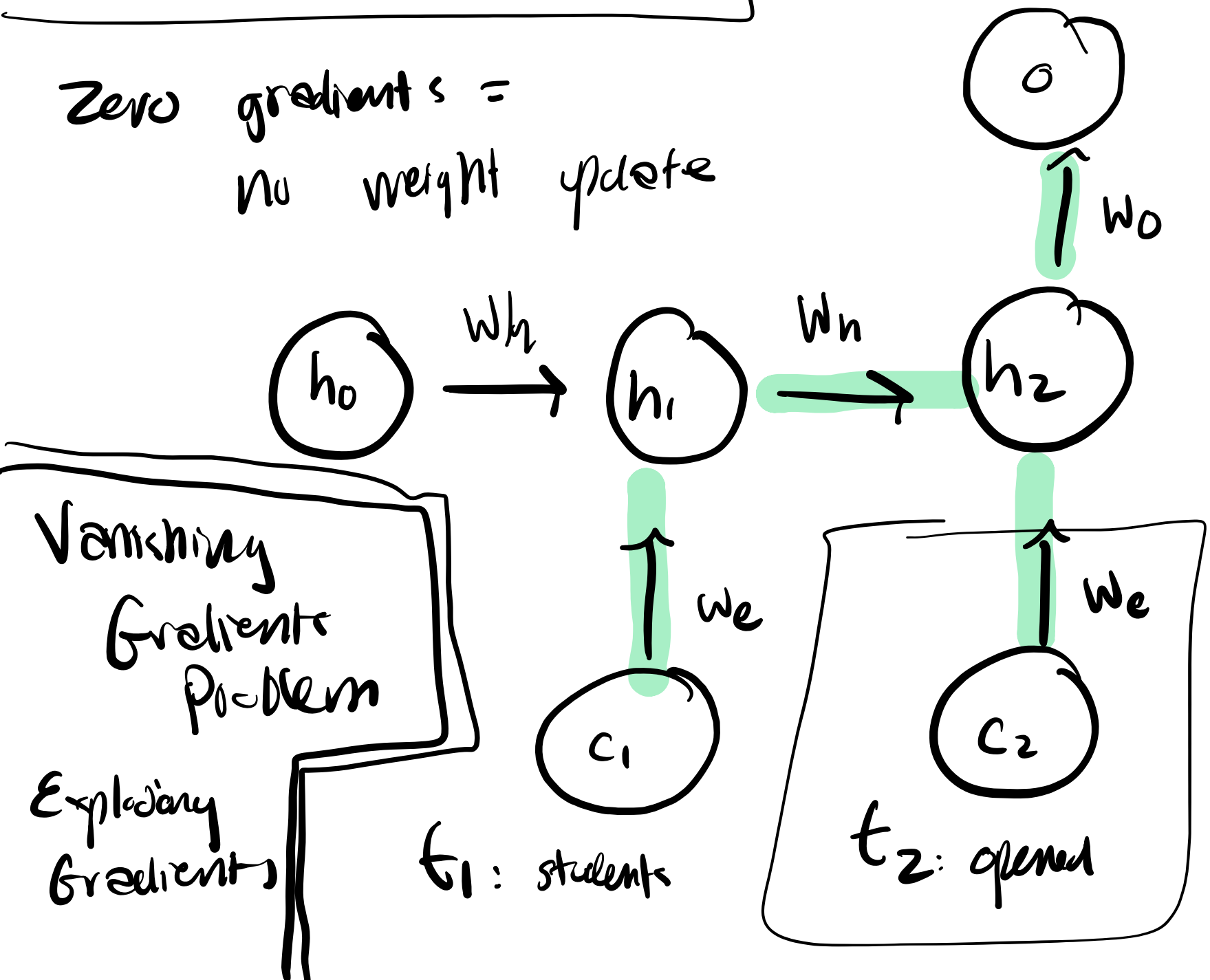
- Recurrent computation is slow
- In practice, difficult to access information from many steps back




Slides adapted from Mohit Iyyer

# Vanishing Gradients Problem

Zero gradients = no weight update



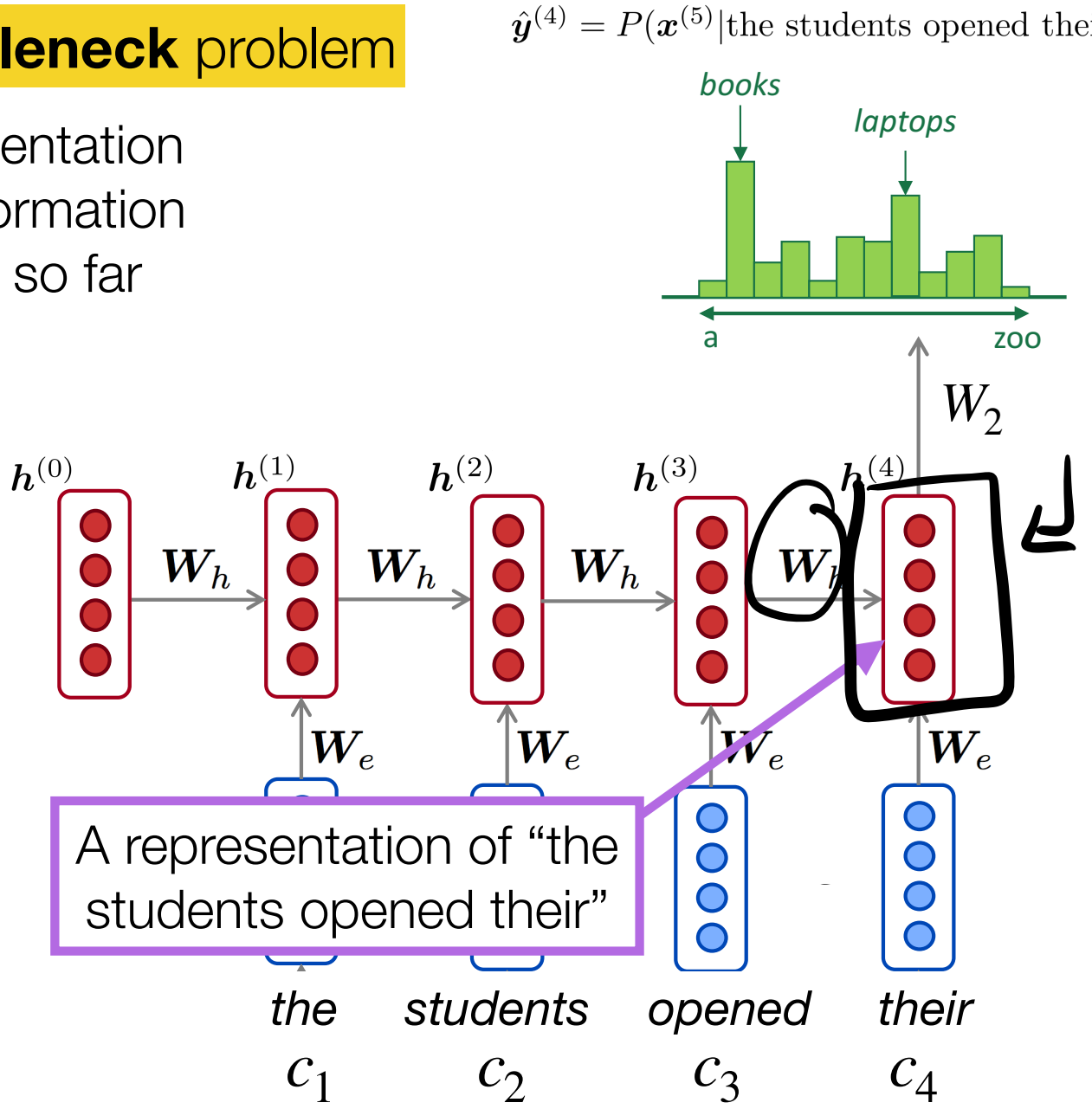

Vanishing Gradients Problem

Exploding Gradients

## RNNs suffer from a **bottleneck** problem

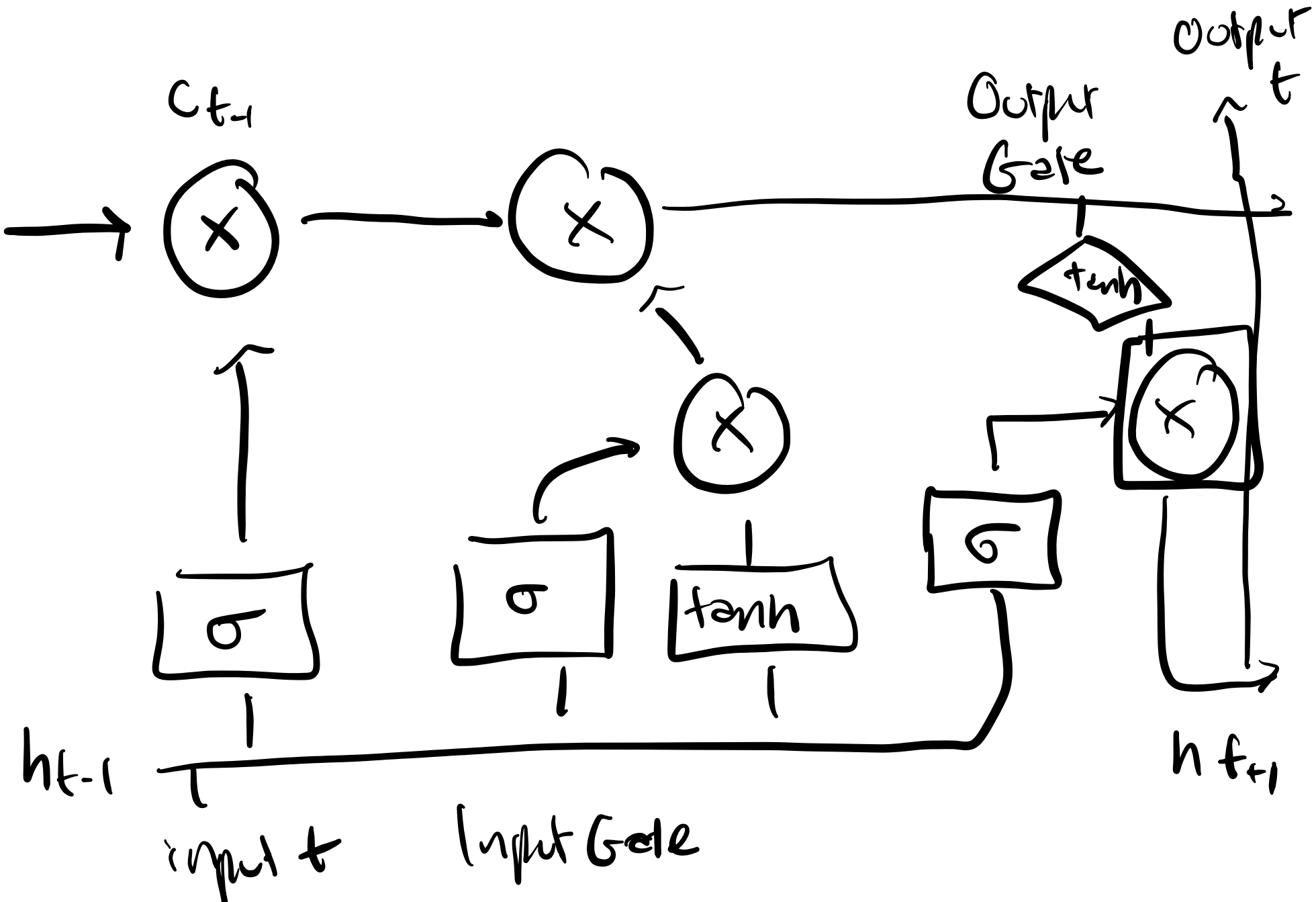
The current hidden representation must encode all of the information about the text observed so far

This becomes difficult especially with longer sequences




Slides adapted from Mohit Iyyer

# LSTMs

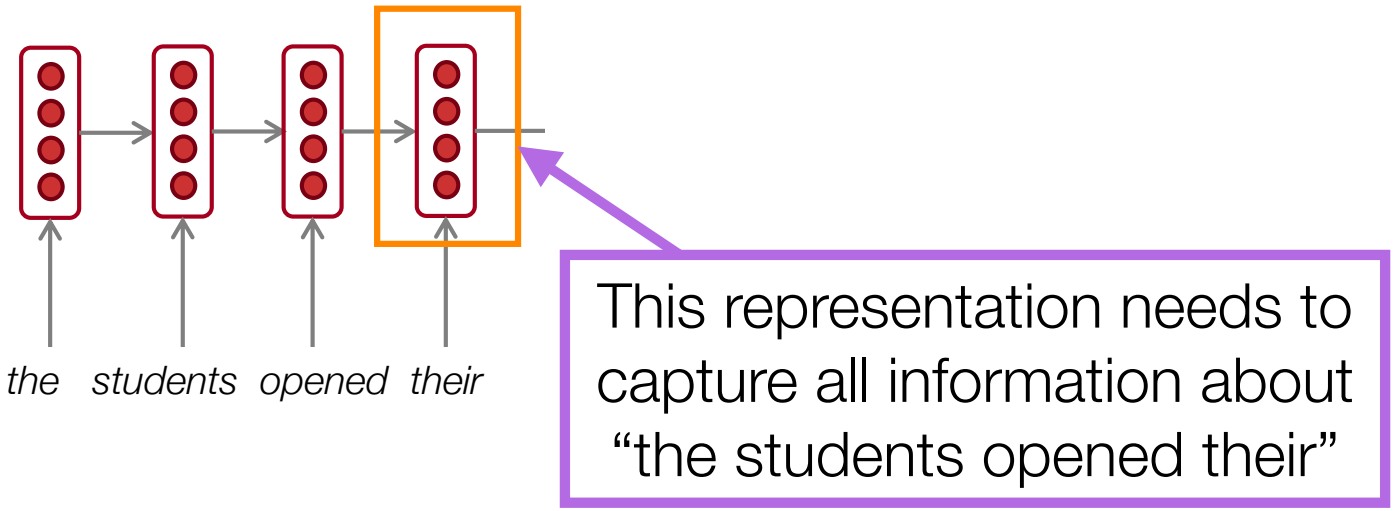

$C_{t-1}$
Output Gate
Output $\hat{t}$
tanh
σ
σ
tanh
σ
$h_{t-1}$
$h_{t+1}$
input $\hat{t}$
Input Gate

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

— Ray Mooney (NLP professor at UT Austin)

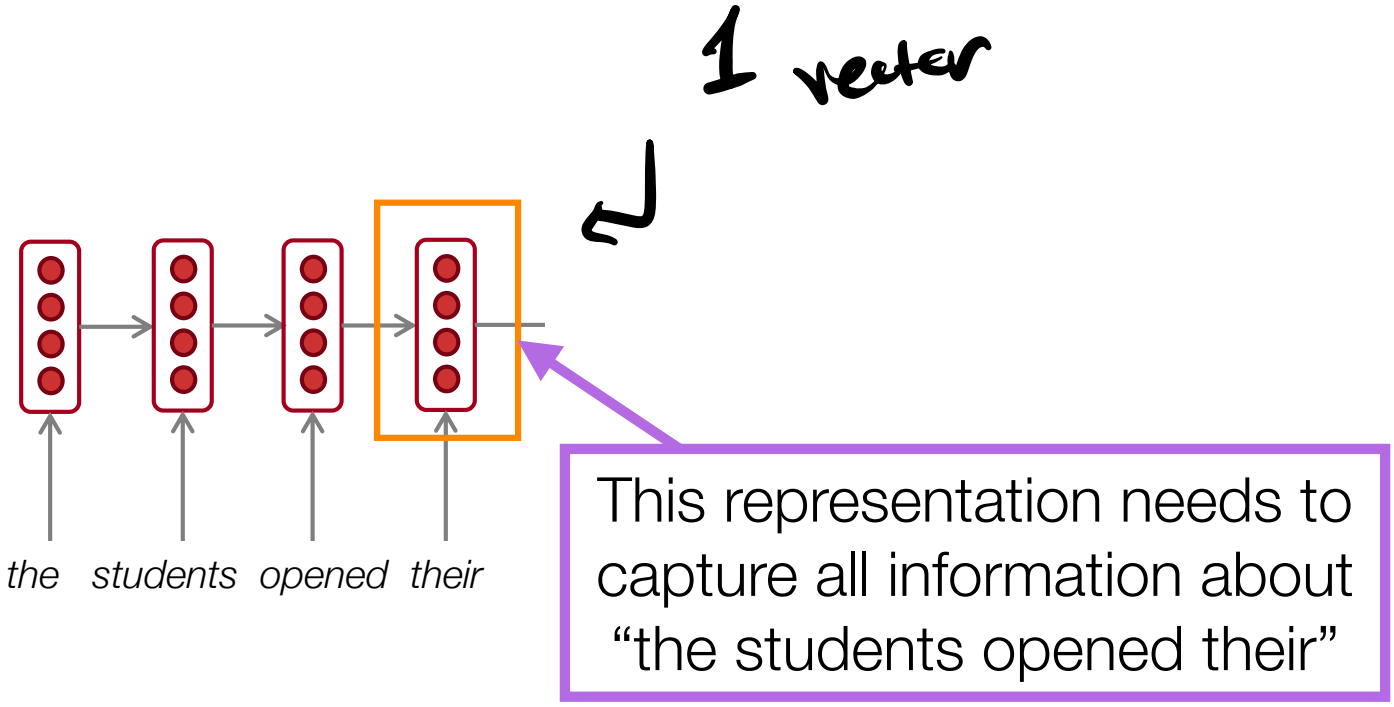
Slides adapted from Mohit Iyyer

# idea: what if we use multiple vectors?

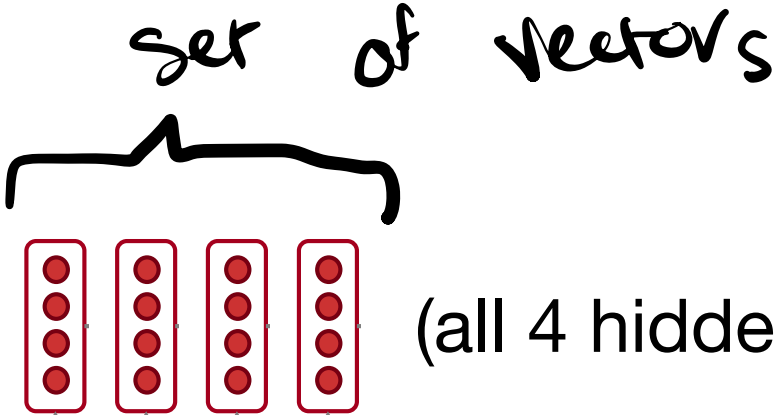



Slides adapted from Mohit Iyyer

# idea: what if we use multiple vectors?




Instead of this, let's try:




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

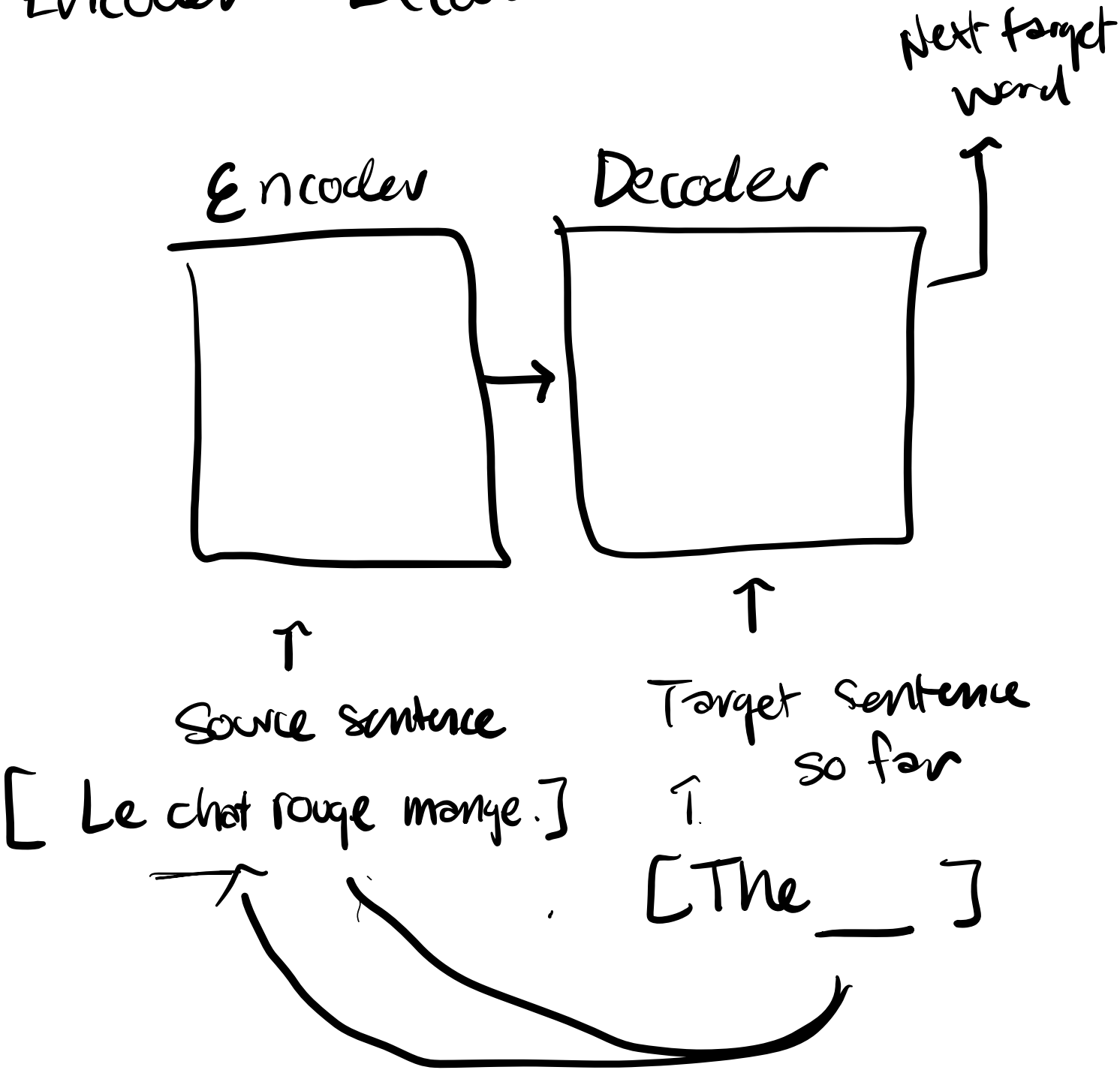
Slides adapted from Mohit Iyyer

# The solution: **attention**

- **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

Slides adapted from Mohit Iyyer

# Encoder – Decoder



# 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 from the past is the most useful?

Slides adapted from Mohit Iyyer

# What Is Attention?

Key: as a task-specific vector
"searching" for important things
from the past context.

Step 1) Measure how similar each x is to k.

dot product

-3.4 $r_1 = k \cdot x_1$ $x_1$: the

2.4 $r_2 = k \cdot x_2$ $x_2$: students

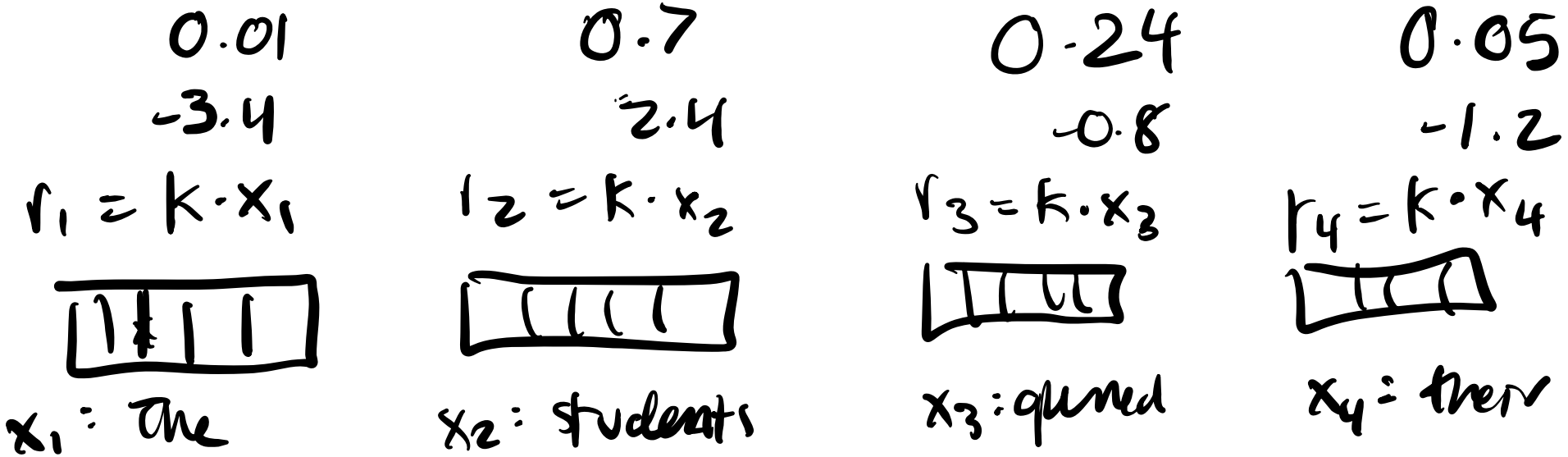
-0.8 $r_3 = k \cdot x_3$ $x_3$: quiet

-1.2 $r_4 = k \cdot x_4$ $x_4$: there

# What Is Attention?

Key: as a task-specific vector
"searching" for important things
from the past context.

Step 2): Scale scores to between 0-1
(softmax)



# What Is Attention?

Key: as a task-specific vector
"searching" for important things
from the past context.

Step 3) Compute a weighted average

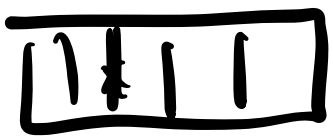
Output: a vector of the same dimension
as a word embedding

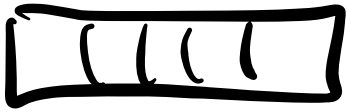
$\Sigma = a x$

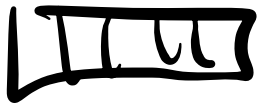
$a = 0.01$ $a = 0.7$ $a = 0.24$ $a = 0.05$

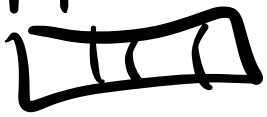
$-3.4$ $2.4$ $-0.8$ $-1.2$

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









$x_1$: the $x_2$: students $x_3$: quiet $x_4$: there

**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:

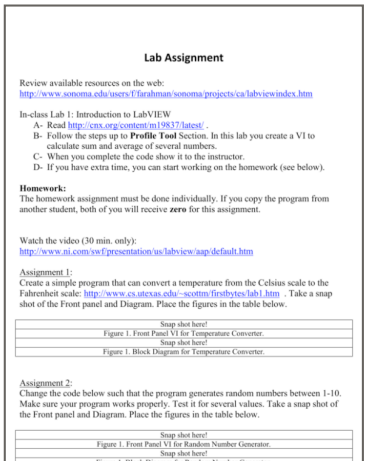
**My brother, a chemist, was late yesterday because he missed the bus. When he arrived, he was surprised to find that his lab _____**

**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

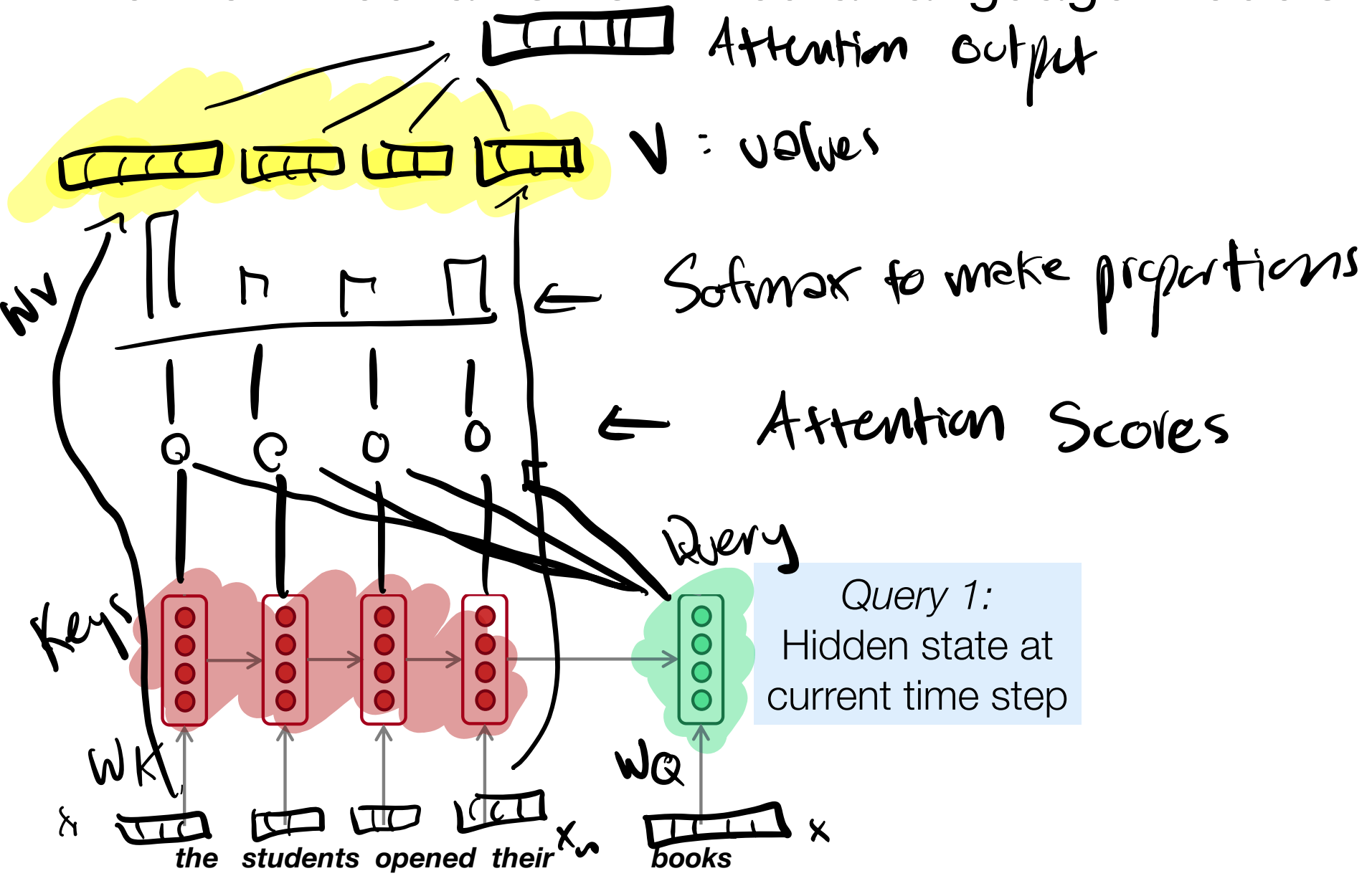
Assignment 1:
Create a simple program that can convert a temperature from the Celsius scale to the Fahrenheit scale: http://www.cs.utexas.edu/~scottm/firstbytes/lab1.htm . Take a snap shot of the Front panel and Diagram. Place the figures in the table below.

| 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.

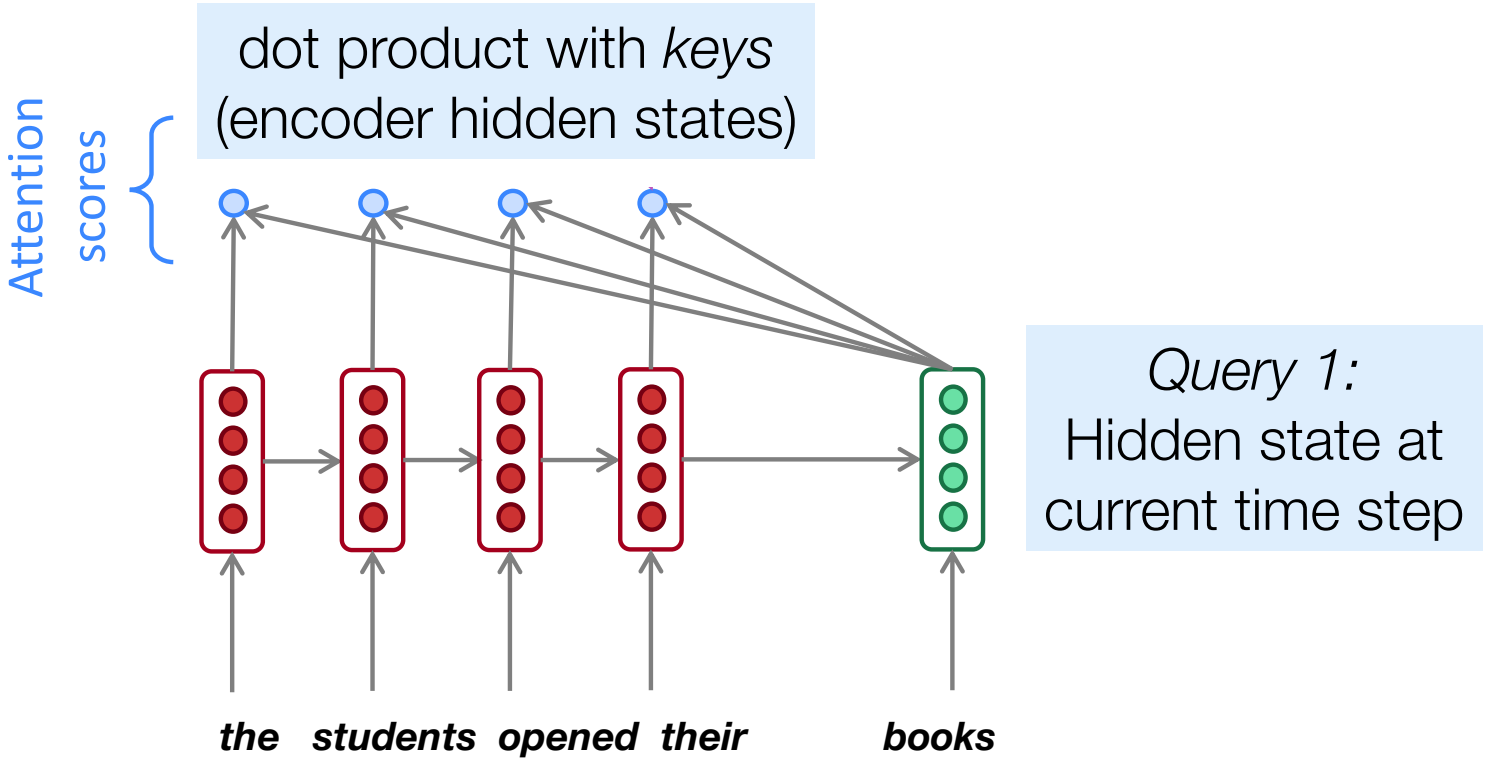
| Snap shot here! |
| --- |
| Figure 1. Front Panel VI for Random Number Generator. |
| Snap shot here! |

# Attention mechanisms in neural language models



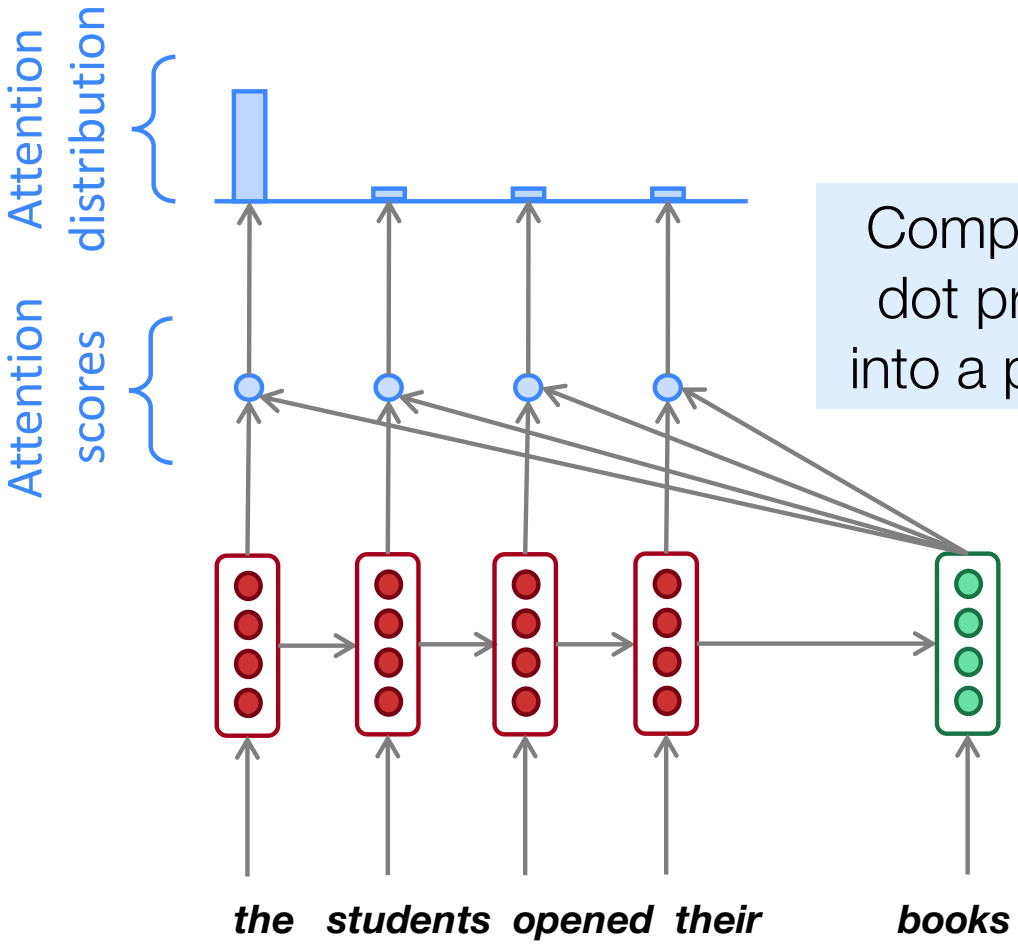

Slides adapted from Mohit Iyyer

# Attention mechanisms in neural language models




Slides adapted from Mohit Iyyer

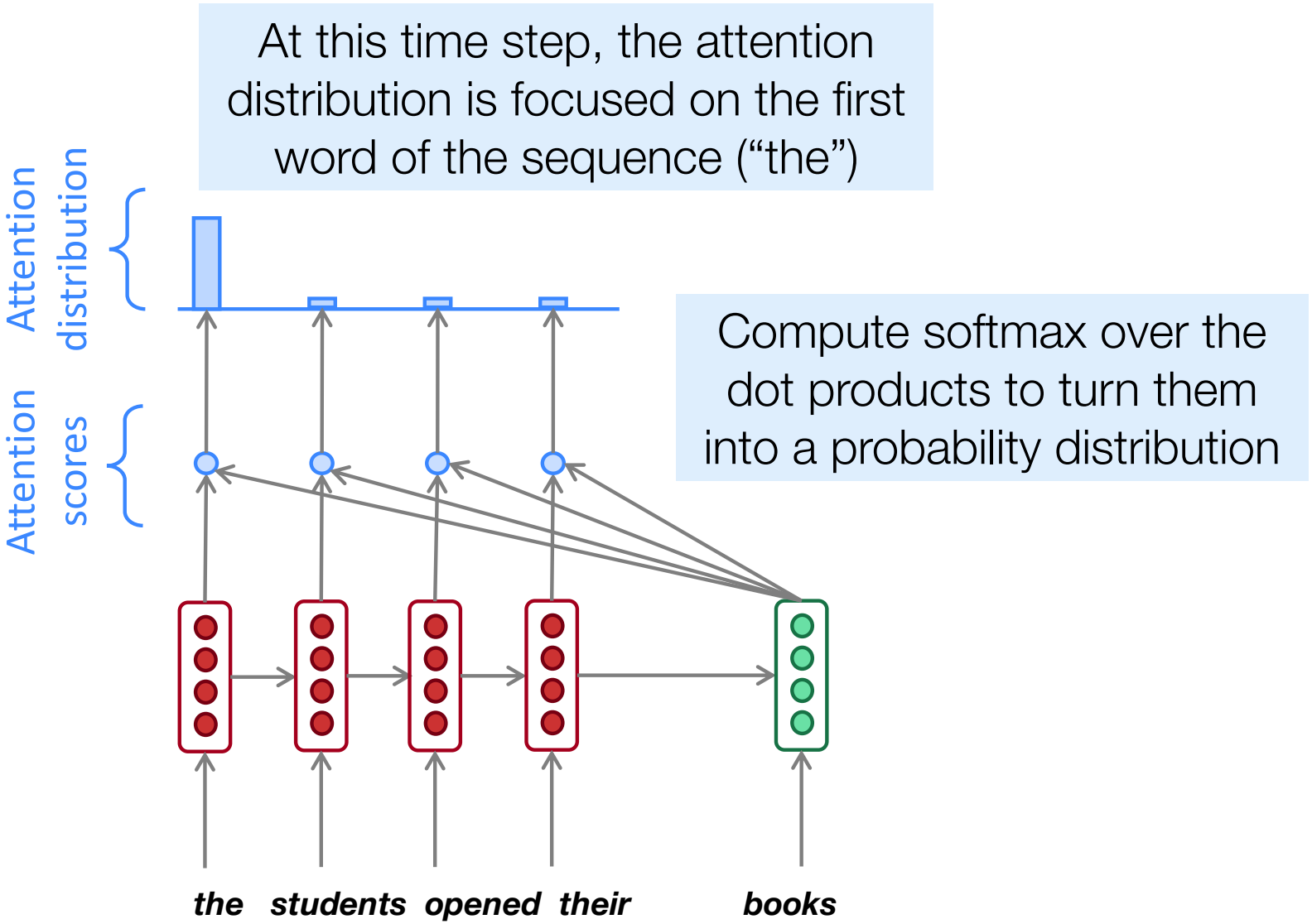
# Attention mechanisms in neural language models




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

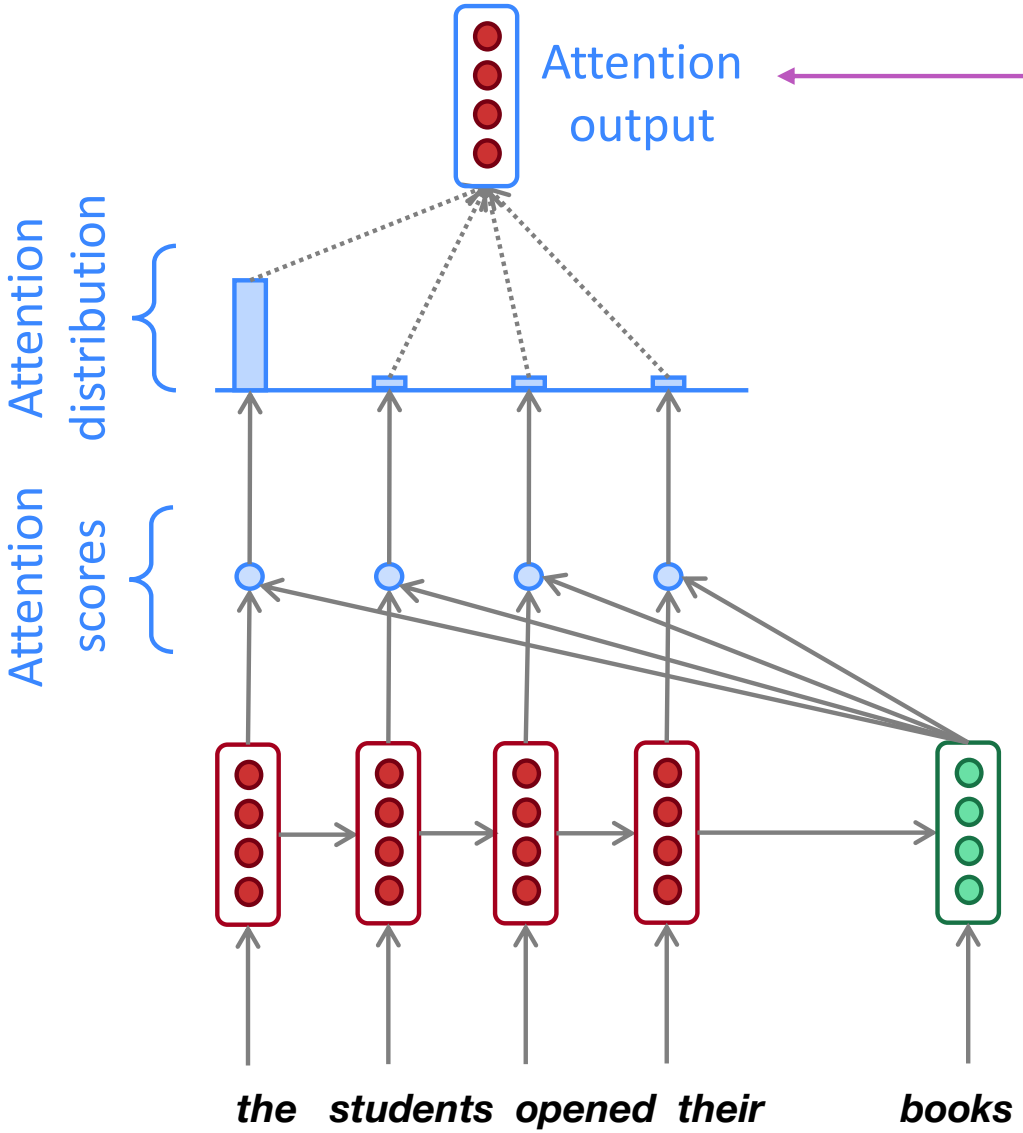
Slides adapted from Mohit Iyyer

# Attention mechanisms in neural language models




Slides adapted from Mohit Iyyer

# Attention mechanisms in neural language models

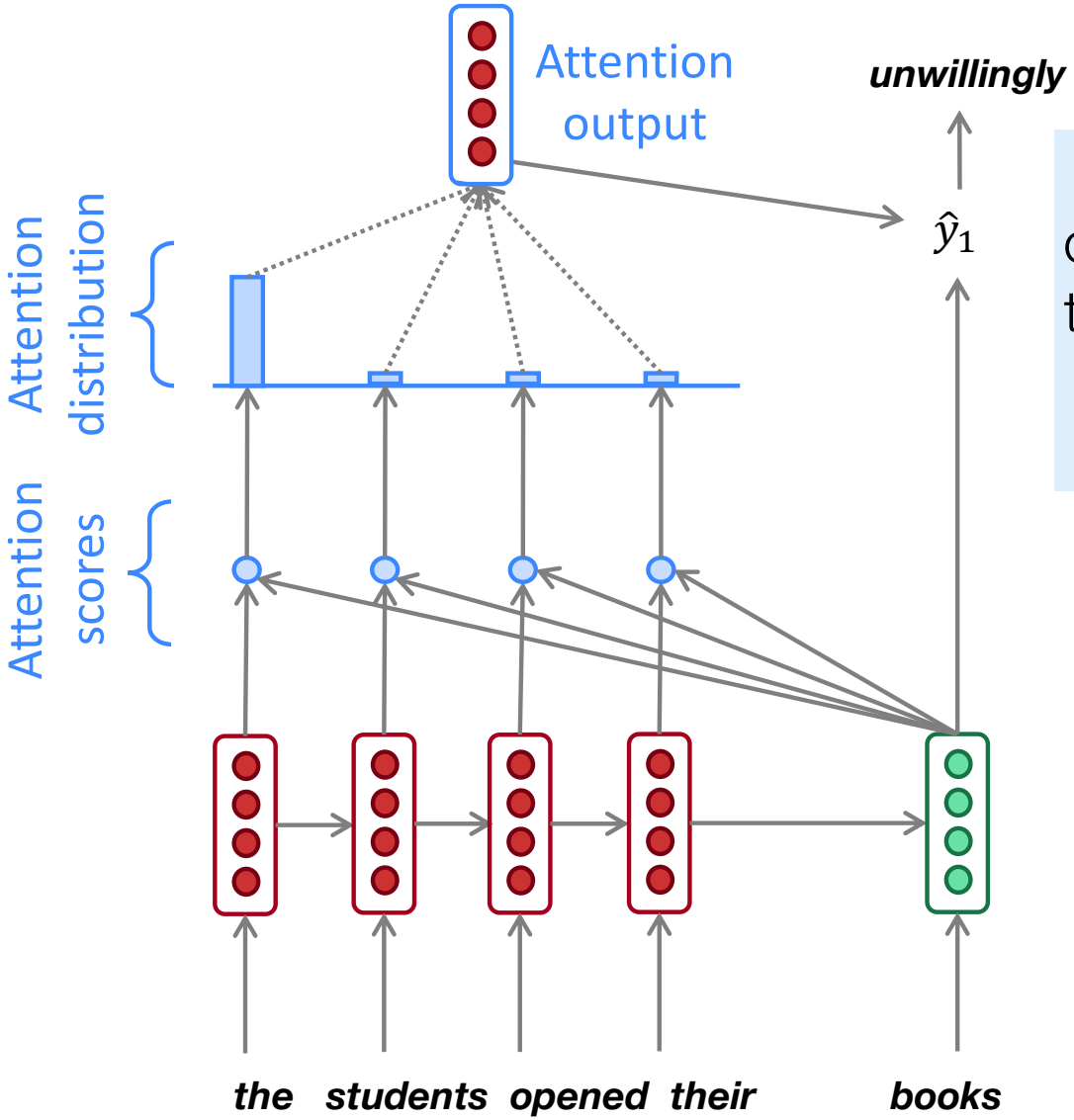



We use the attention distribution to compute a weighted average of the hidden states.

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

Slides adapted from Mohit Iyyer

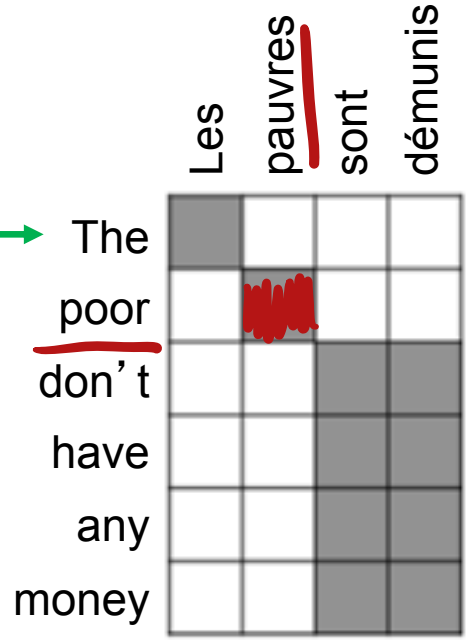
# Sequence-to-sequence with attention




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

- 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 adapted from Mohit Iyyer

# Many variants of attention

- Original formulation: $a(\mathbf{q}, \mathbf{k}) = w_2^T \tanh(W_1[\mathbf{q}; \mathbf{k}])$
- Bilinear product: $a(\mathbf{q}, \mathbf{k}) = \mathbf{q}^T W \mathbf{k}$ Luong et al., 2015
- Dot product: $a(\mathbf{q}, \mathbf{k}) = \mathbf{q}^T \mathbf{k}$ Luong et al., 2015
- Scaled dot product: $a(\mathbf{q}, \mathbf{k}) = \frac{\mathbf{q}^T \mathbf{k}}{\sqrt{|\mathbf{k}|}}$ Vaswani et al., 2017

Slides adapted from Mohit Iyyer

# Self-Attention

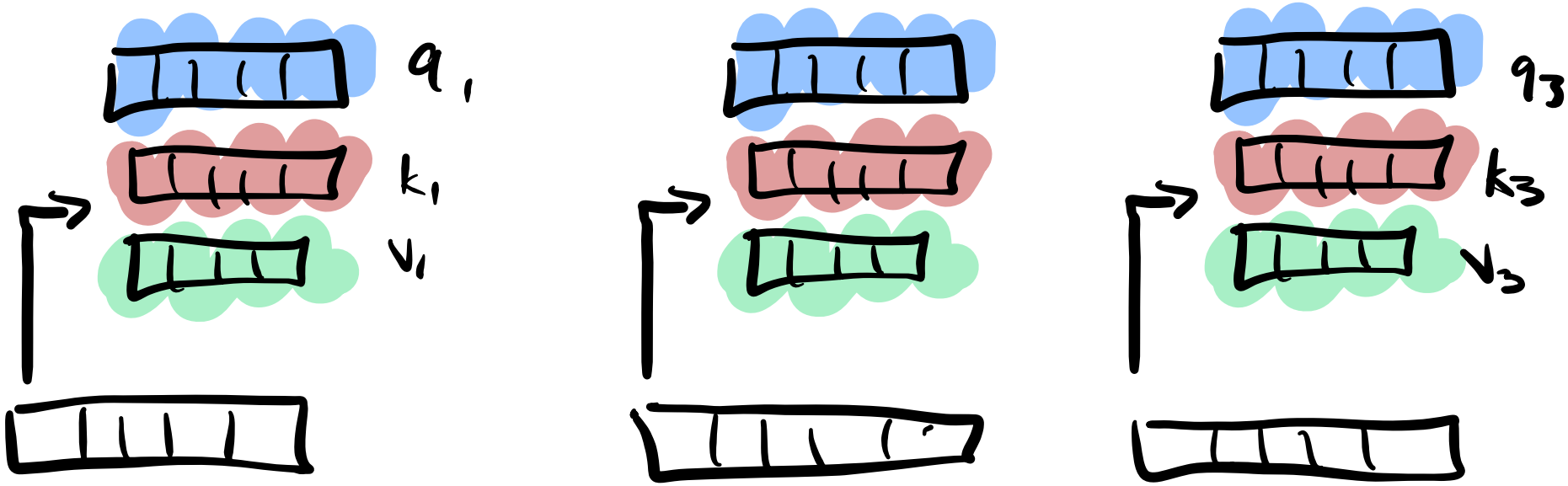
# Self-Attention

Motivation: efficiency by parallelization

$q_1 = f(W_q c_1)$ $k_1 = f(W_k c_1)$ $v_1 = f(W_v c_1)$

1. Take the dot product between $q_3$ & every $k$

$< q_3 k_1 \quad q_3 k_2 \quad q_3 k_3 >$

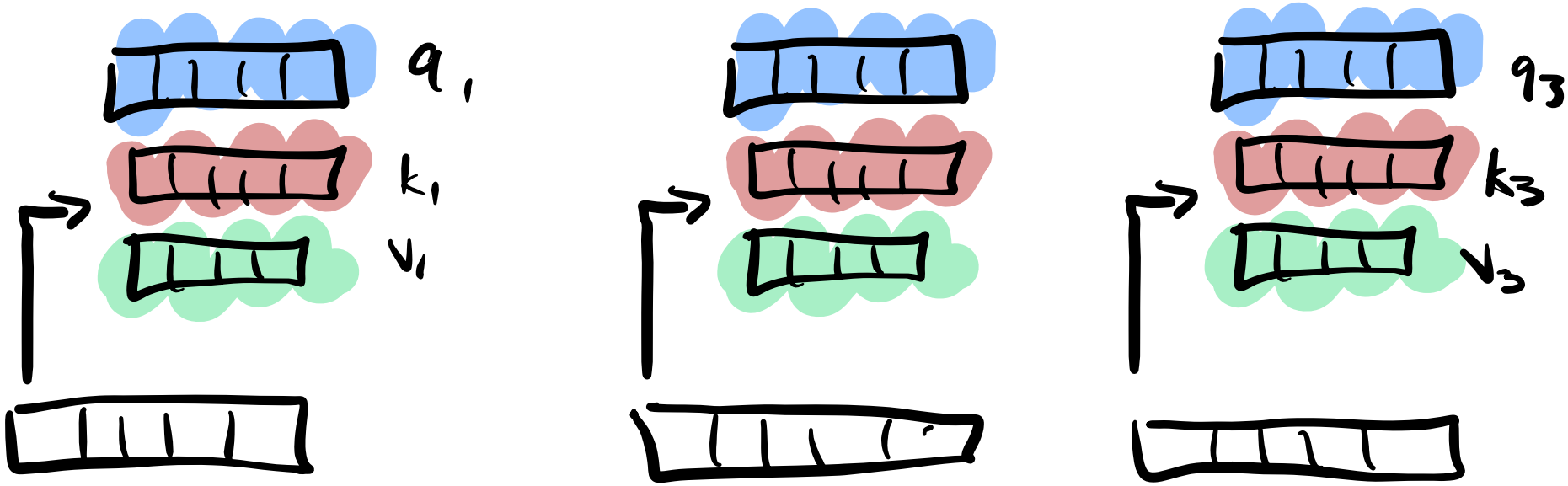



$c_1$: students $c_2$: opened $c_3$: their

Self-Attention Motivation: efficiency & parallelization

Step 2): Softmax to get a distribution


$\langle 0.3 \; 0.5 \; 0.2 \rangle$




$C_1$: Students

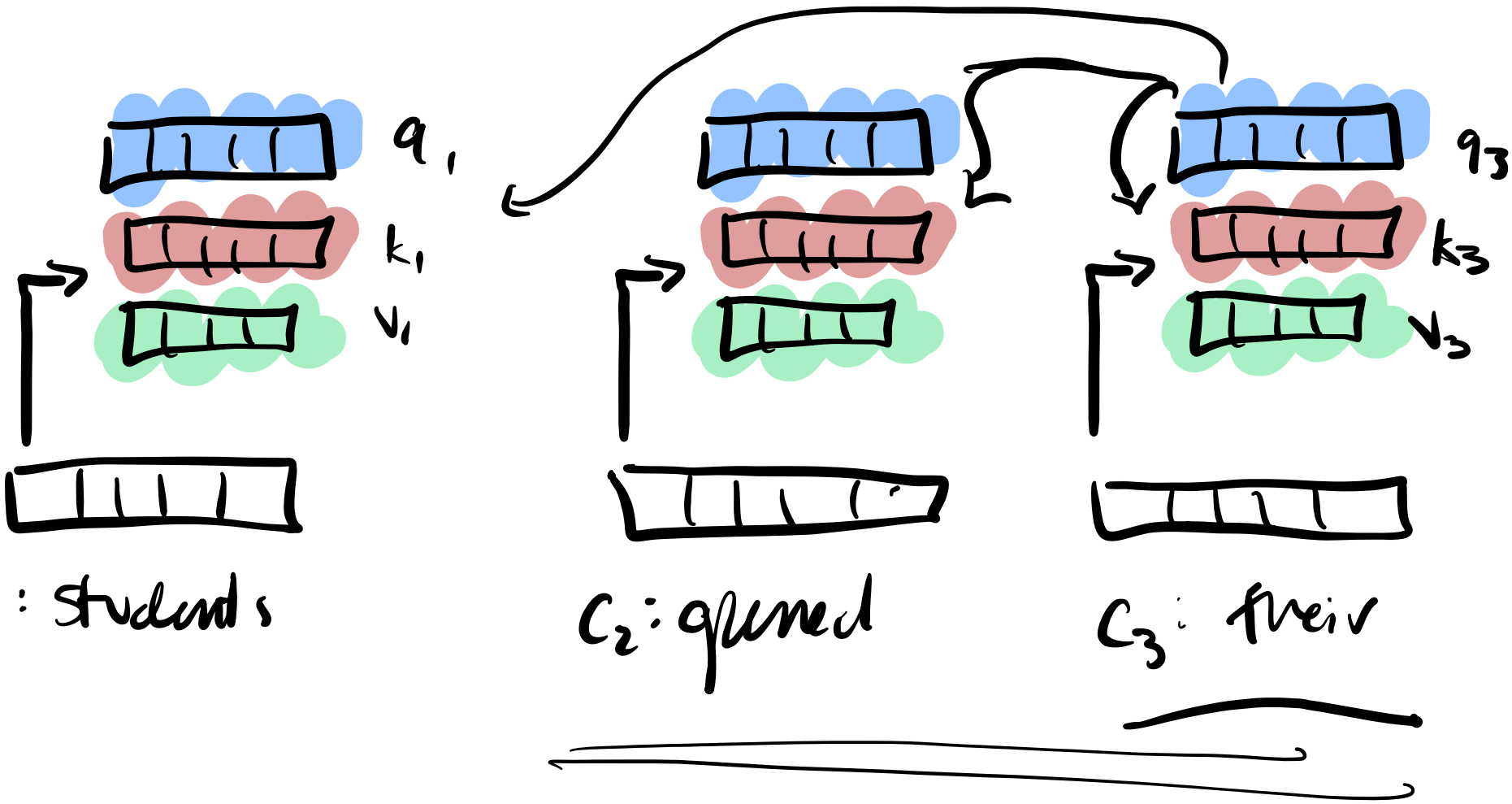
$C_2$: opened

$C_3$: their

Self-Attention Motivation: efficiency & parallelization

step 3): Calculate weighted average on values.

$$h_3 = 0.3v_1 + 0.5v_2 + 0.2v_3$$



Self-Attention Motivation: efficiency & parallelization

step 3): Calculate weighted average on values.

must be zoo!

$h_3 = 0.3v_1 + 0.5v_2 + 0.2v_3$