Department of Linguistics and Translation
香港城市大學
City University of Hong Kong

Computational Linguistics LT3233

Jixing Li

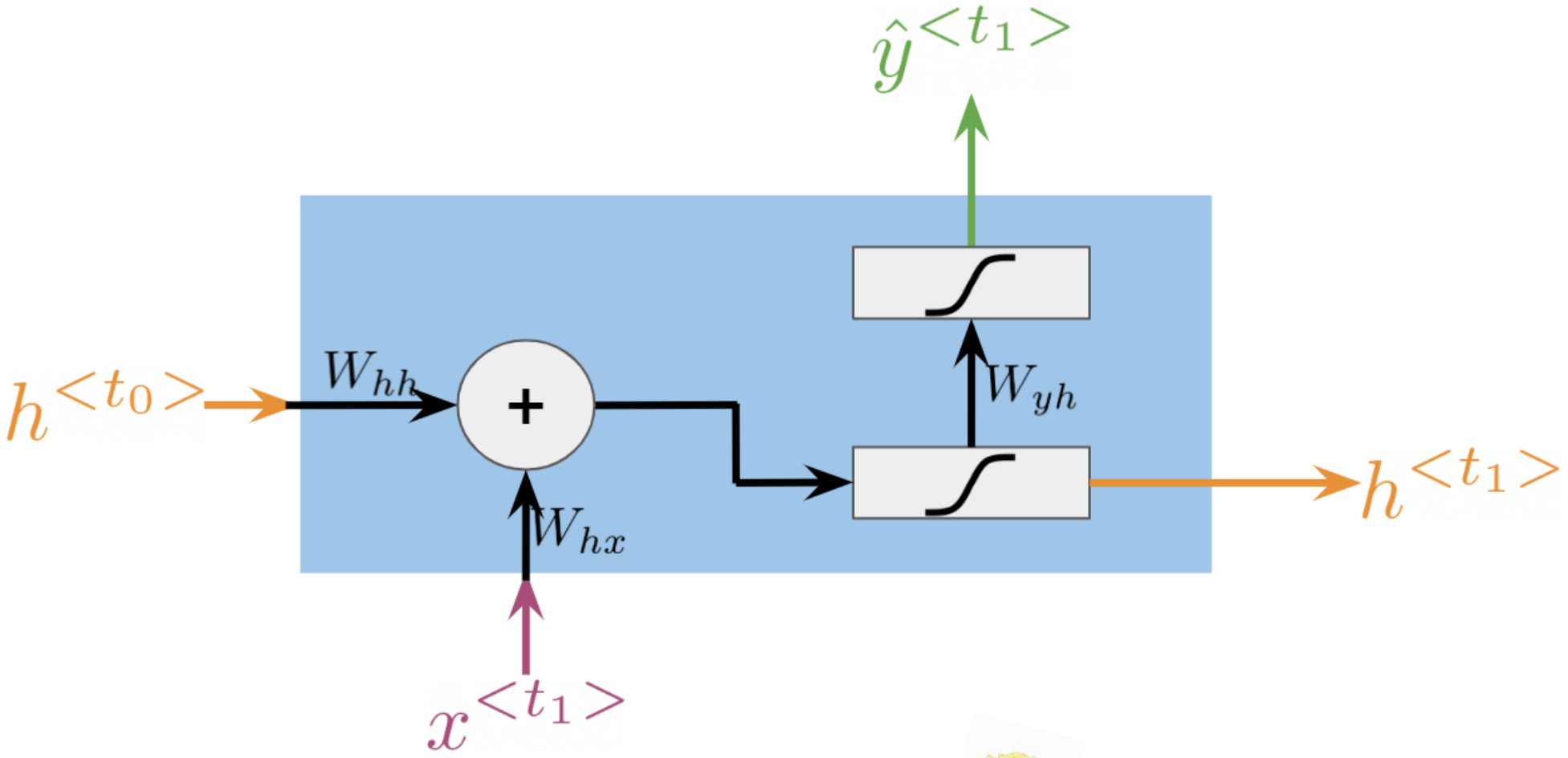
Lecture 13: Attention and Transformers

Slides adapted from Dan Jurafsky, Ellie Pavlick, Anna Goldie, Jay Alammar

Lecture plan

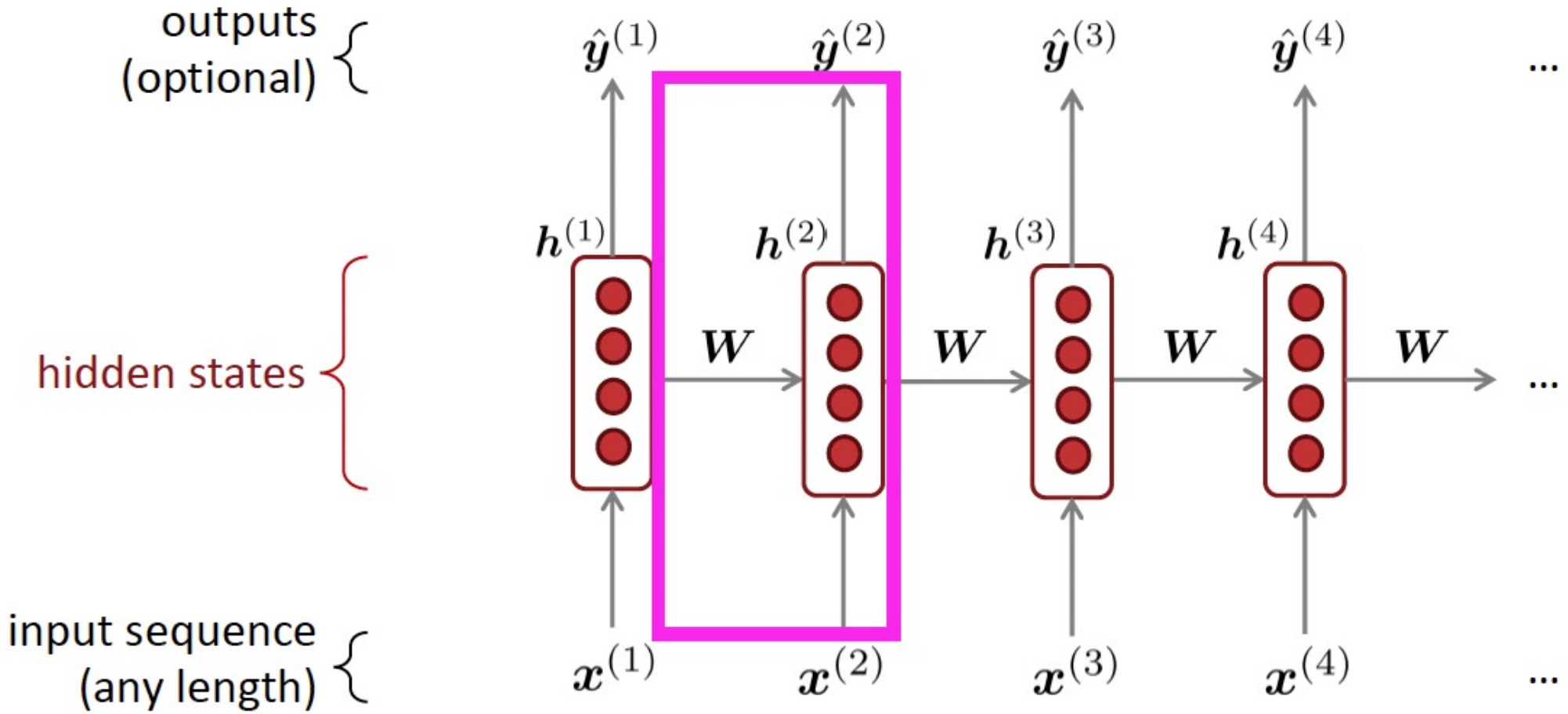
- Problems with RNN
- Attention
- Transformers
- Short break (15 mins)
- Hands-on exercises

Recurrent Neural Networks (RNN)



simple/vanilla/Elman RNN

RNN basic structure



RNN language model

output distribution

$$\hat{\boldsymbol{y}}^{(t)} = \text{softmax}\left(\boldsymbol{U}\boldsymbol{h}^{(t)} + \boldsymbol{b}_2\right) \in \mathbb{R}^{|V|}$$

hidden states

$$\boldsymbol{h}^{(t)} = \sigma\left(\boldsymbol{W}_h\boldsymbol{h}^{(t-1)} + \boldsymbol{W}_e\boldsymbol{e}^{(t)} + \boldsymbol{b}_1\right)$$

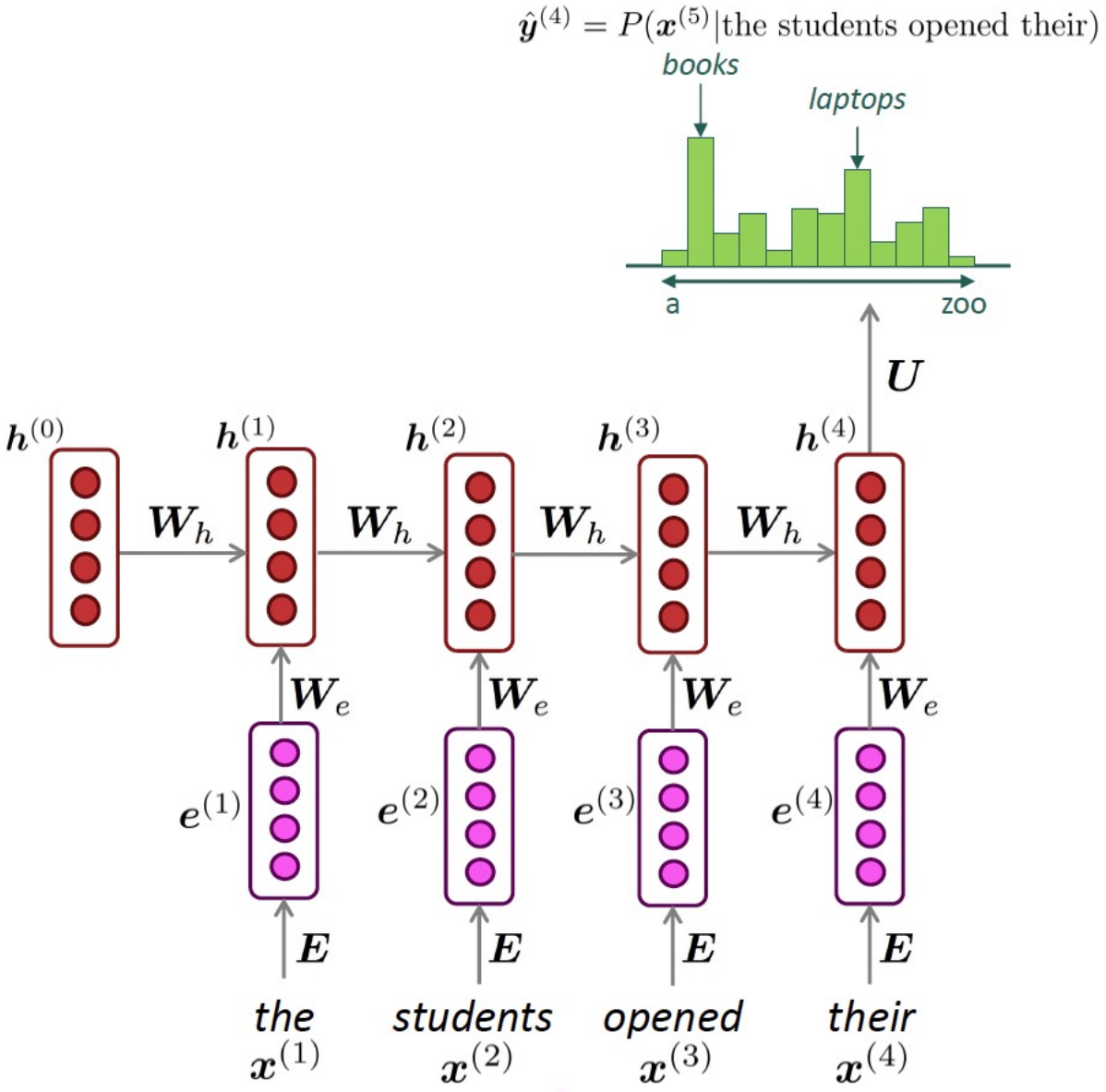
$\boldsymbol{h}^{(0)}$ is the initial hidden state

word embeddings

$$\boldsymbol{e}^{(t)} = \boldsymbol{E}\boldsymbol{x}^{(t)}$$

words / one-hot vectors

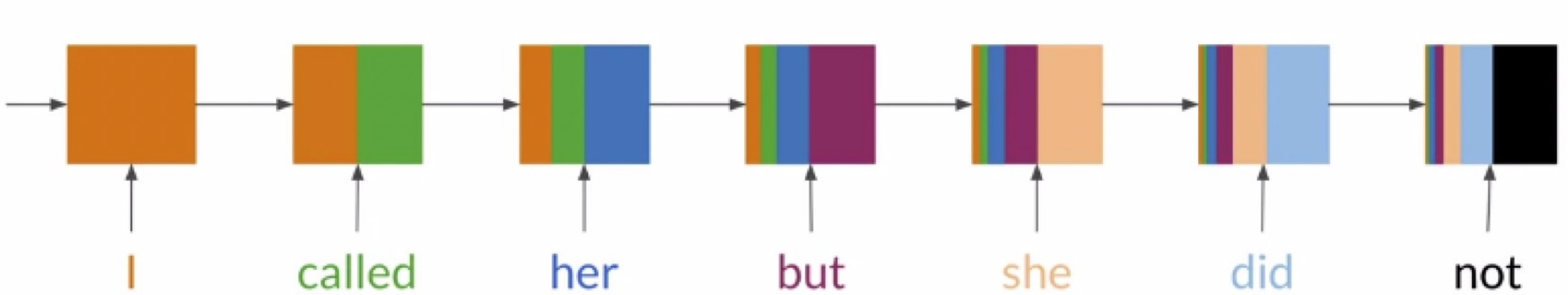
$$\boldsymbol{x}^{(t)} \in \mathbb{R}^{|V|}$$



RNN language model

Advantages:

- Can process any length input
- Computation for step *t* can (in theory) use information from many steps back
- Model size doesn't increase for longer input context: Same weights applied on every timestep

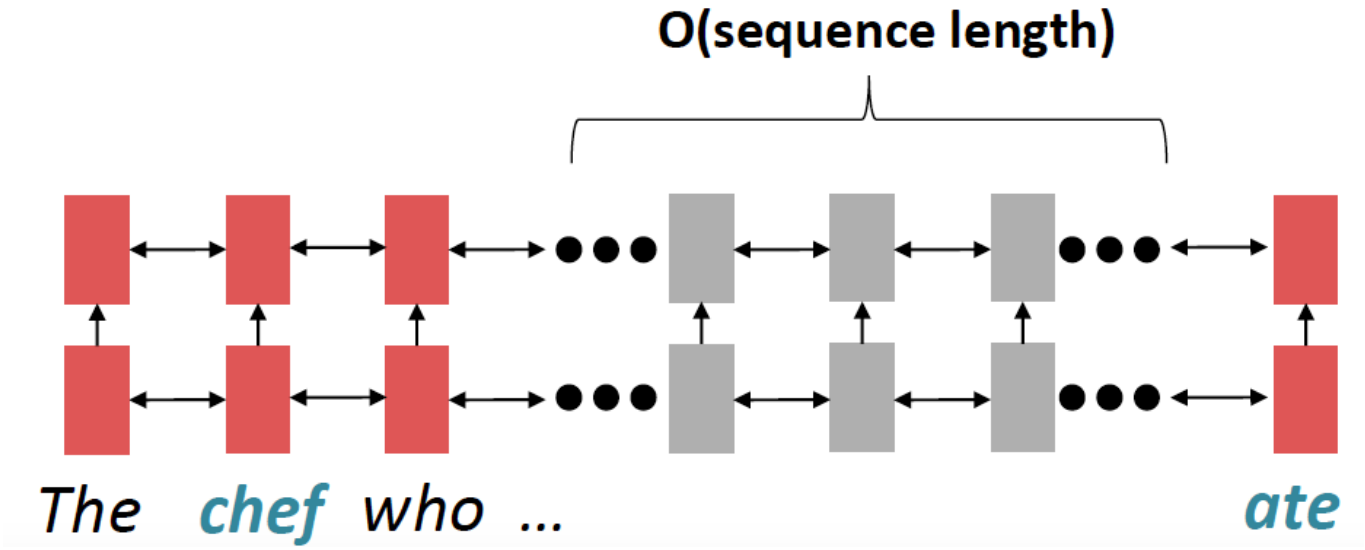


RNN language model

Disadvantages:

RNNs take **O(sequence length)** steps for distant word pairs to interact.

- Hard to learn long-distance dependencies (vanishing gradients)
- Forward and backward passes have ***O*** unparallelizable operations:
  - future RNN hidden states can't be computed before past RNN hidden states have been computed



Info of ***chef*** has gone through O(sequence length) many layers!

Attention

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

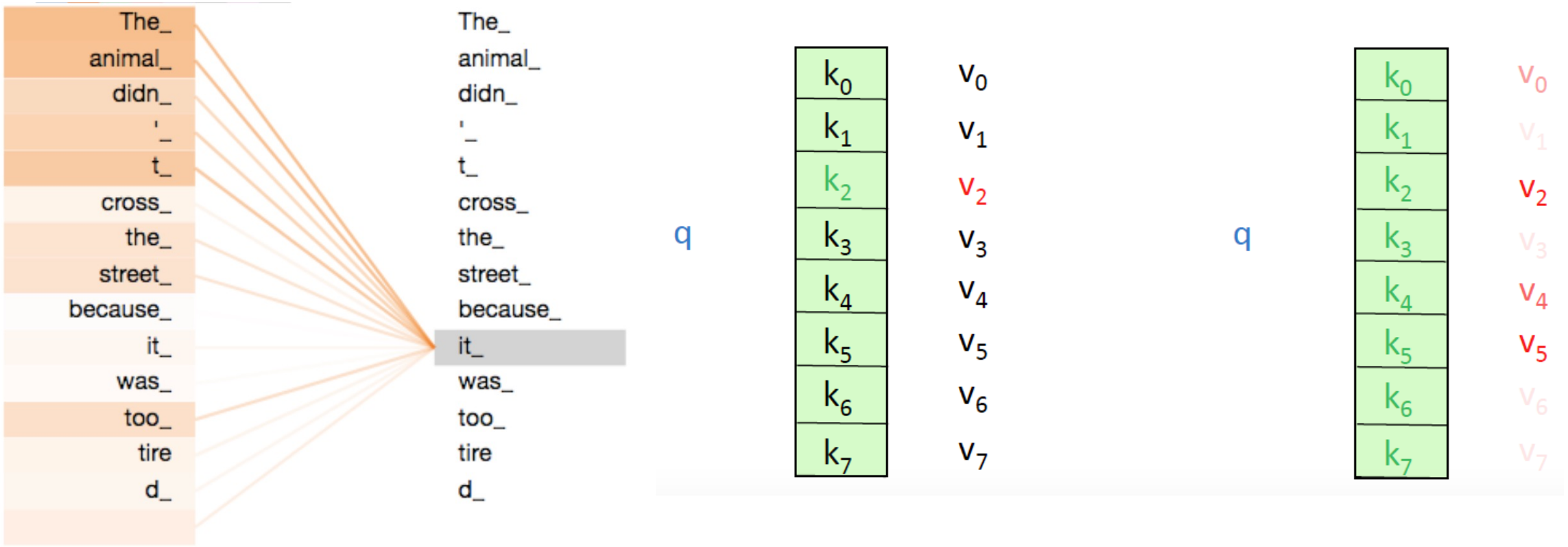
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Idea: Compare an item of interest to a collection of other items in a way that reveals their relevance in the current context.

Attention is the core innovation in **Transformer** architecture which revolutionizes deep learning

Attention intuition

"The animal didn't cross the street because it was too tired."



To look up a **value**, we compare a **query** against **keys** in a table.
Each query matches each key to varying degrees.
We return a sum of values weighted by the query-key match.

Self-Attention

1. For each word x_i, calculate its query, key, and value.

$$q_i = W^Q x_i \quad k_i = W^K x_i \quad v_i = W^V x_i$$

2. Calculate attention score between query and keys

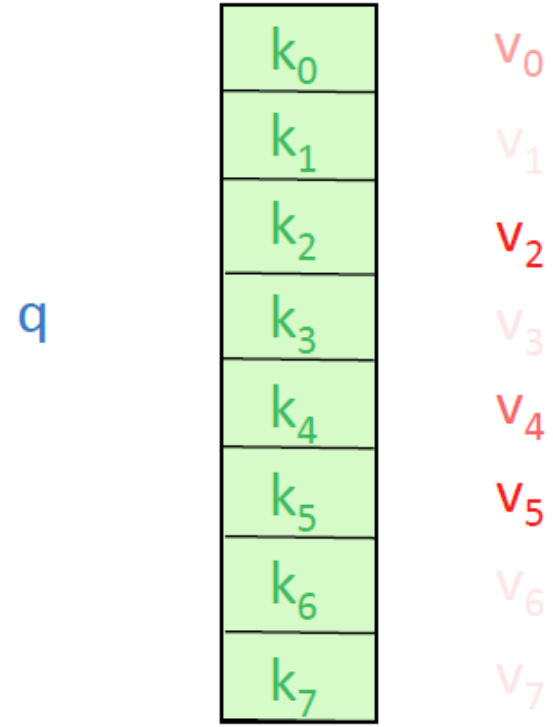
$$e_{ij} = q_i \cdot k_j$$

3. Take the softmax to normalize attention scores.

$$\alpha_{ij} = softmax(e_{ij}) = \frac{exp(e_{ij})}{\sum_k exp(e_{ik})}$$

4. Take a weighted sum of values.

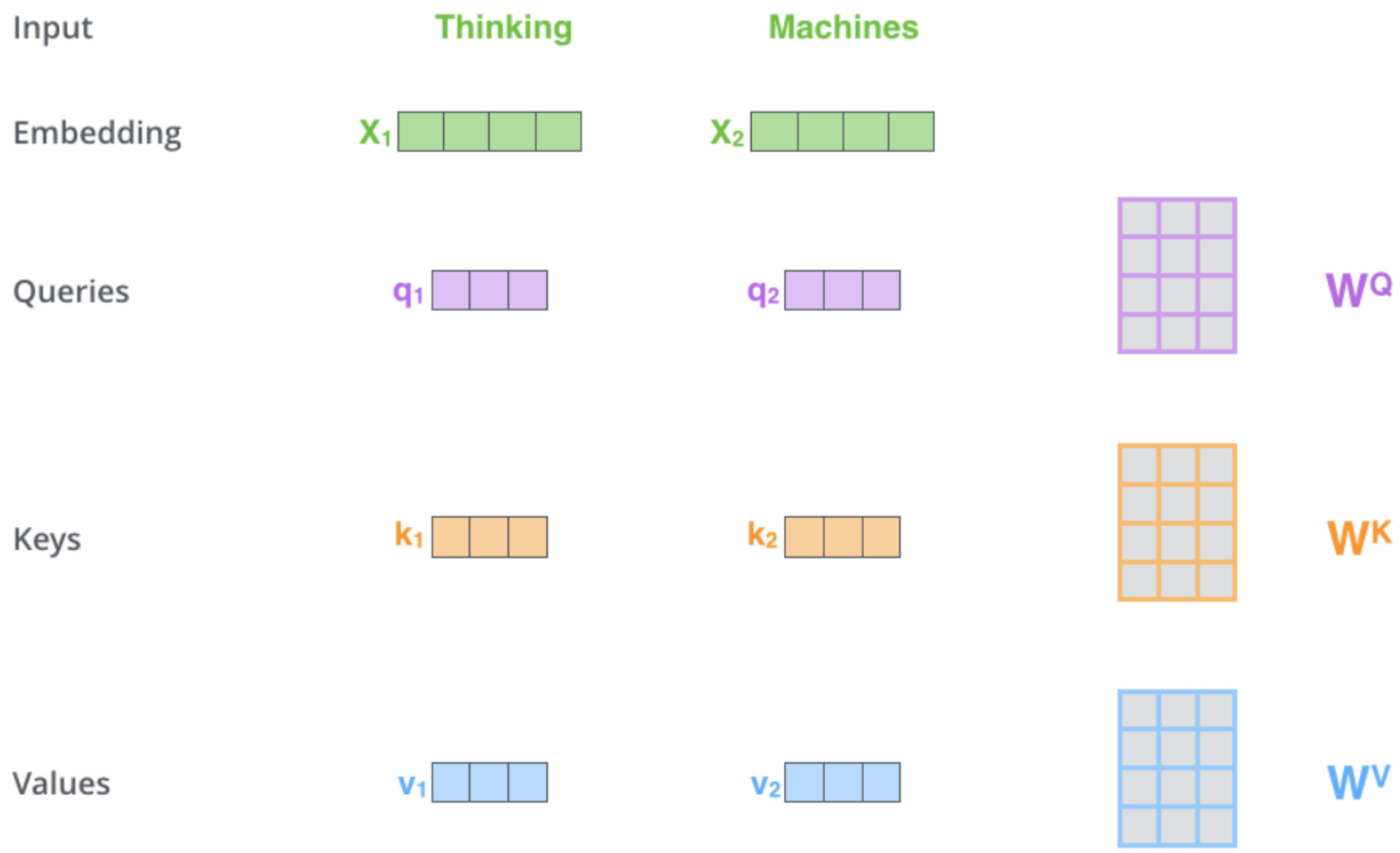
$$Output_i = \sum_j \alpha_{ij} v_j$$



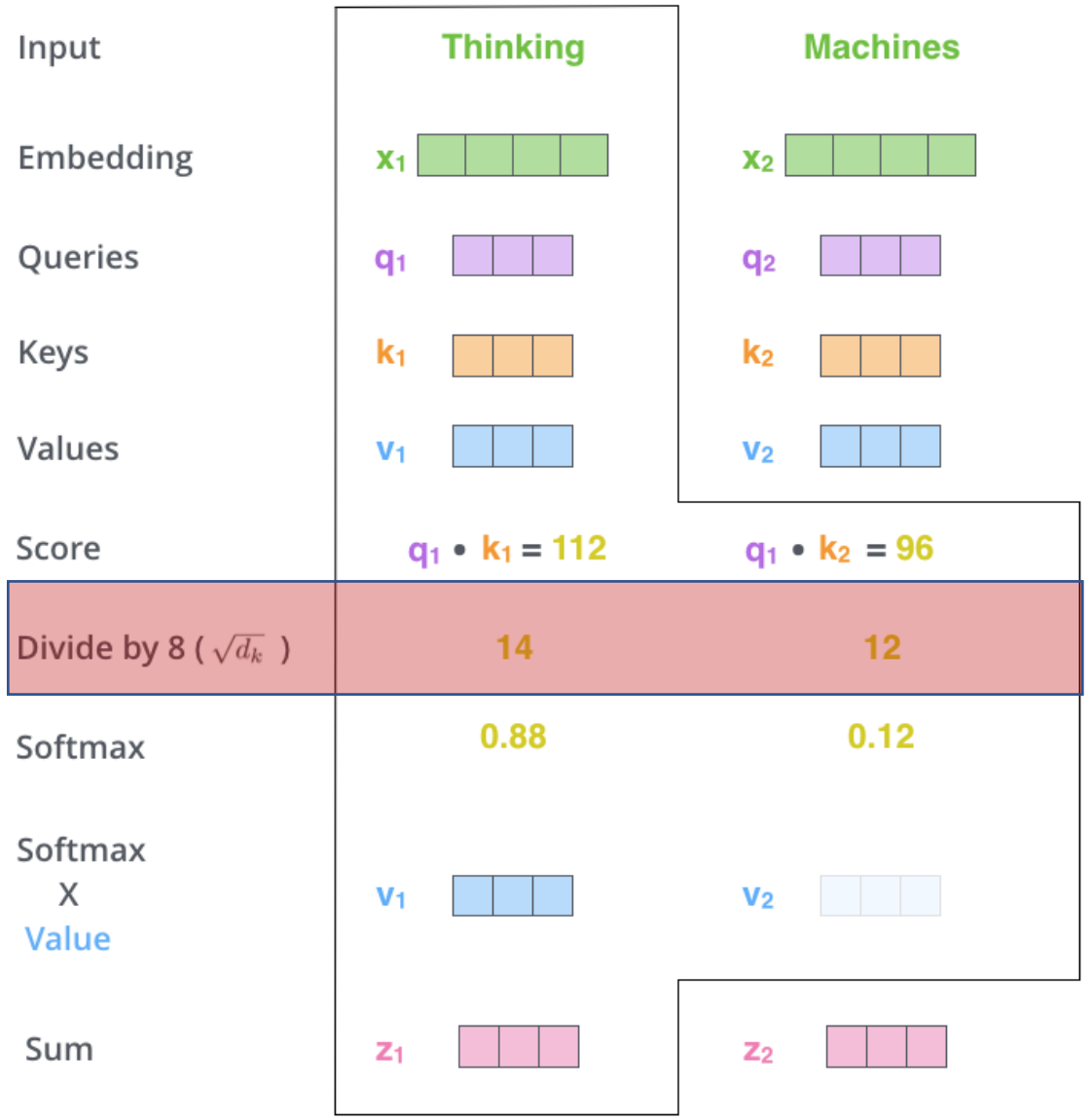
Illustrated self-attention

For each word x_i, calculate its query, key, and value.

$$q_i = W^Q x_i \quad k_i = W^K x_i \quad v_i = W^V x_i$$



Illustrated self-attention



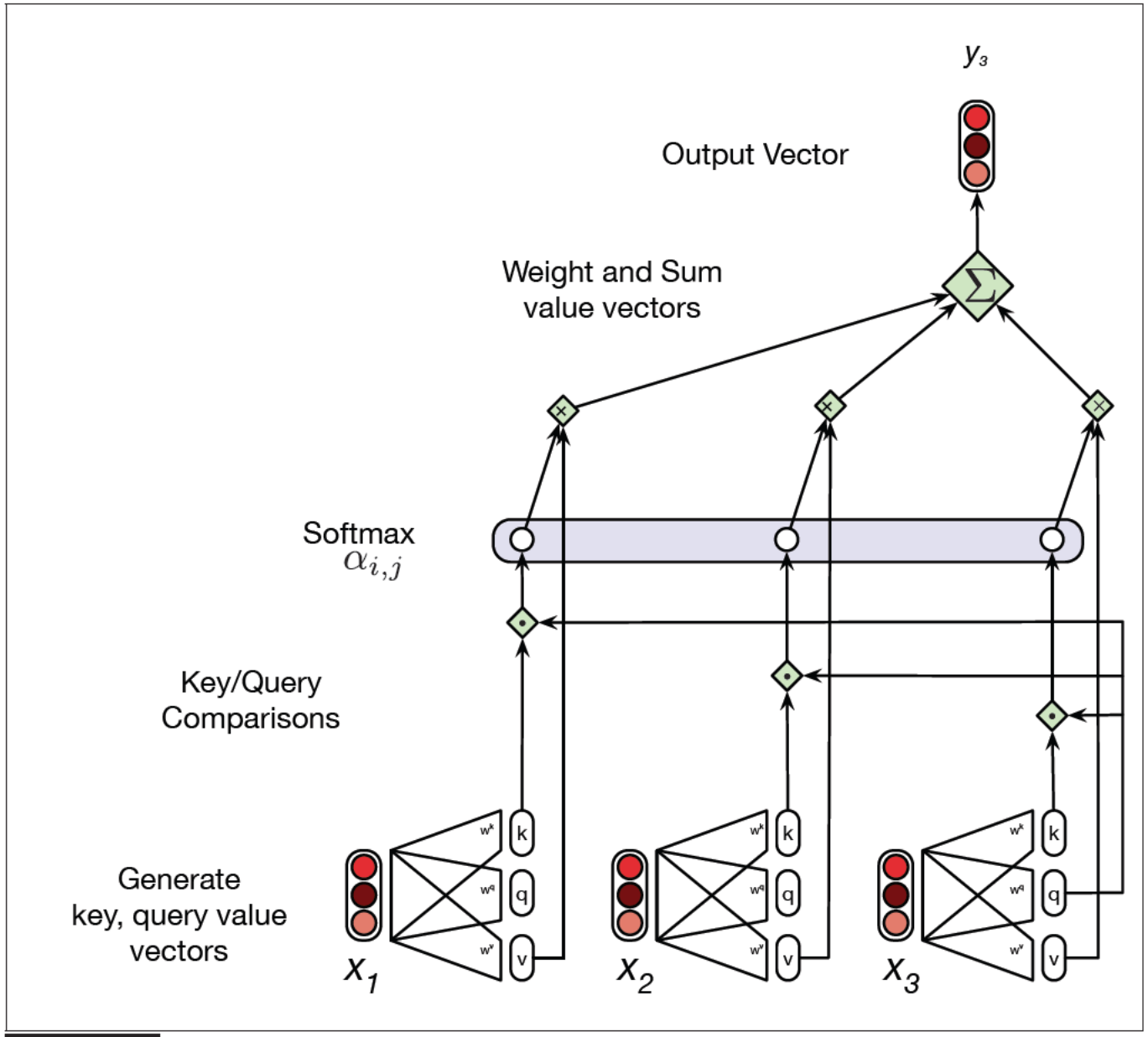
Calculate attention score between query and keys. $e_{ij} = q_i \cdot k_j$

Take the softmax to normalize attention scores.

Take a weighted sum of values.

$$Output_i = \sum_j \alpha_{ij} v_j$$

Illustrated self-attention

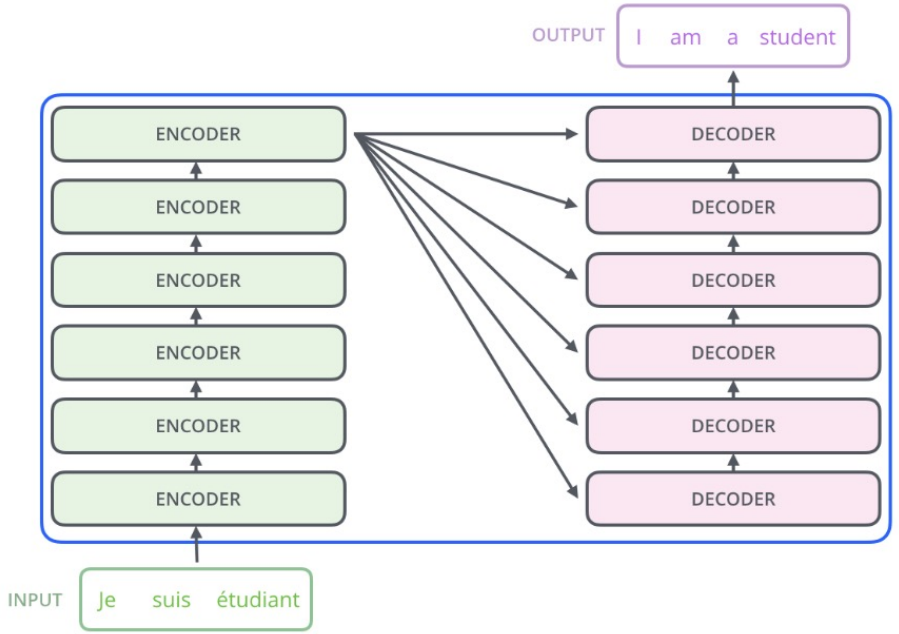


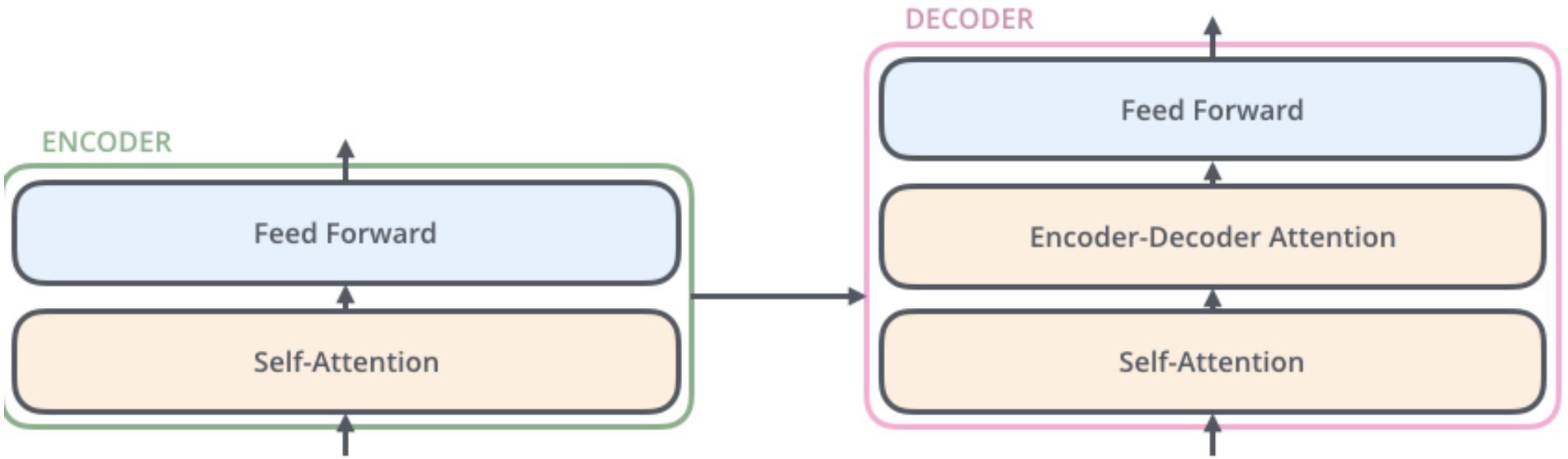
Calculating the value of y_3, the third element of a sequence using self-attention.

Summary

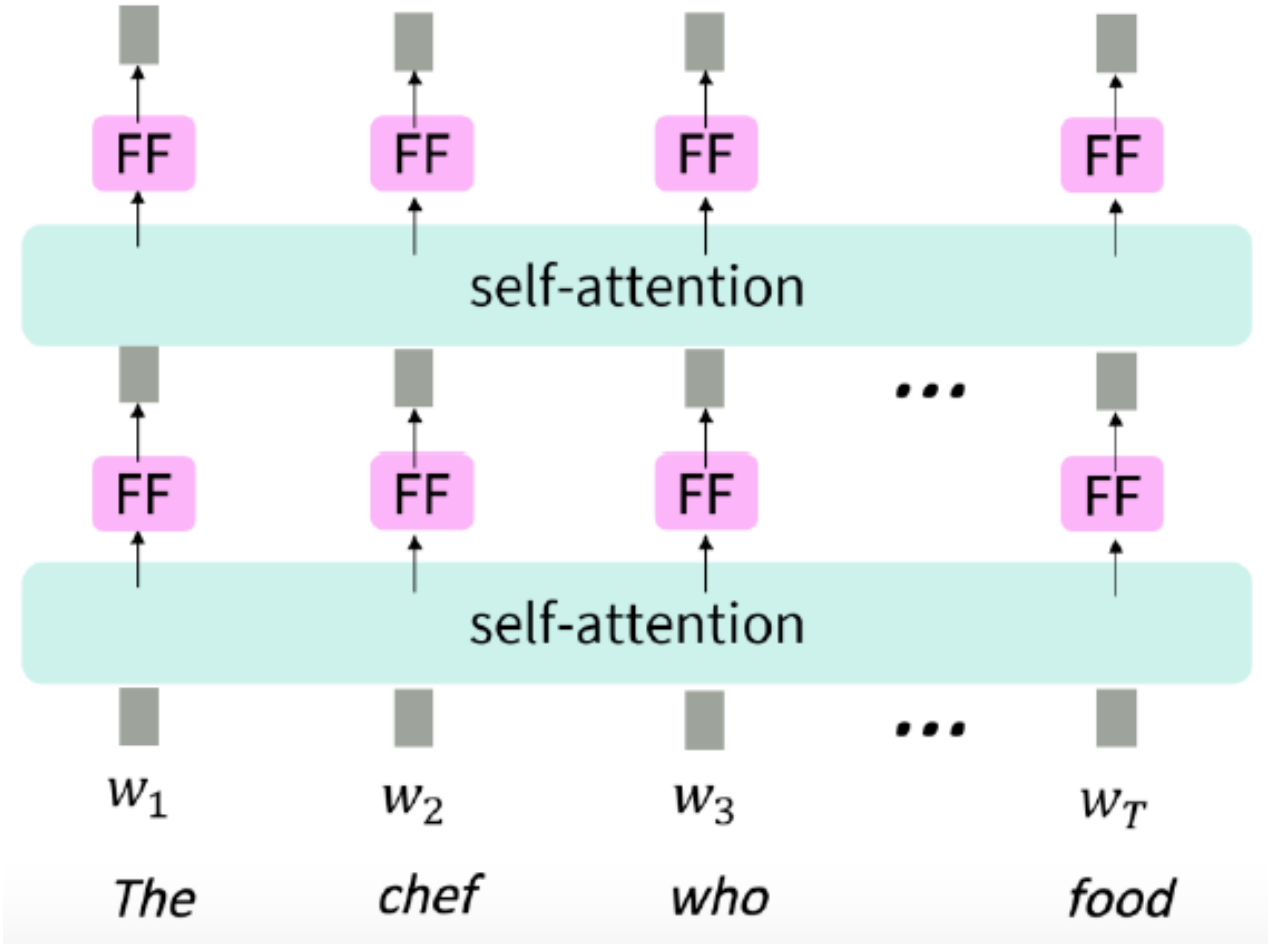
- Attention is an added layer that lets a model focus on what's important
- **Queries, Values, and Keys** are used for information retrieval inside the Attention layer
- Attention has become the powerful, flexible, general way pointer and memory manipulation in all deep learning models.

Transformer





Feed-forward layer

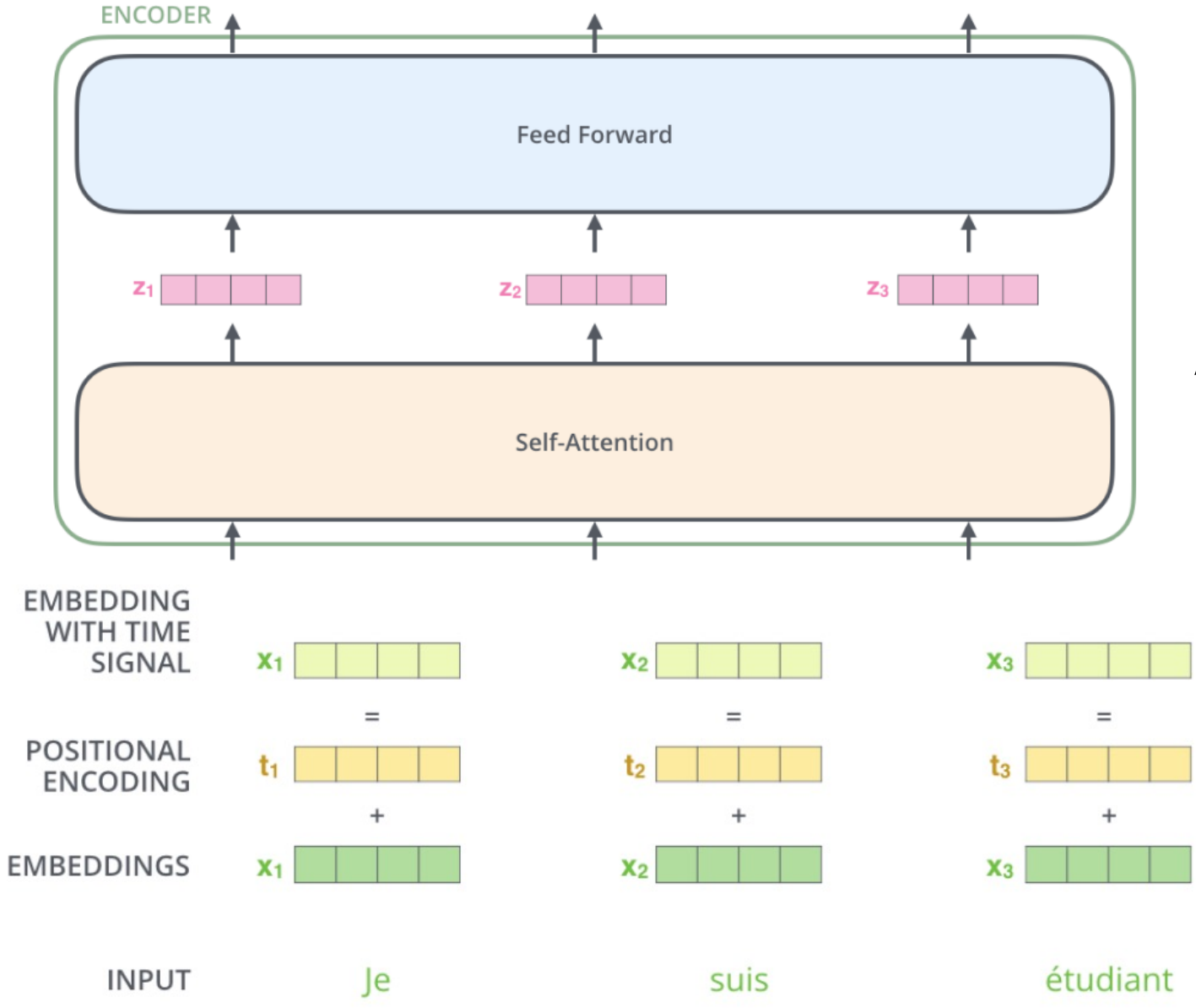


Since there are no element-wise non-linearities, self-attention is simply performing a re-averaging of the value vectors.

→ Apply a feedforward layer to the output of attention, providing non-linear activation (and additional expressive power).

$$W_2 * \mathrm{ReLU}(W_1 \times \mathrm{output}_i + b_1) + b_2$$

Positional embeddings



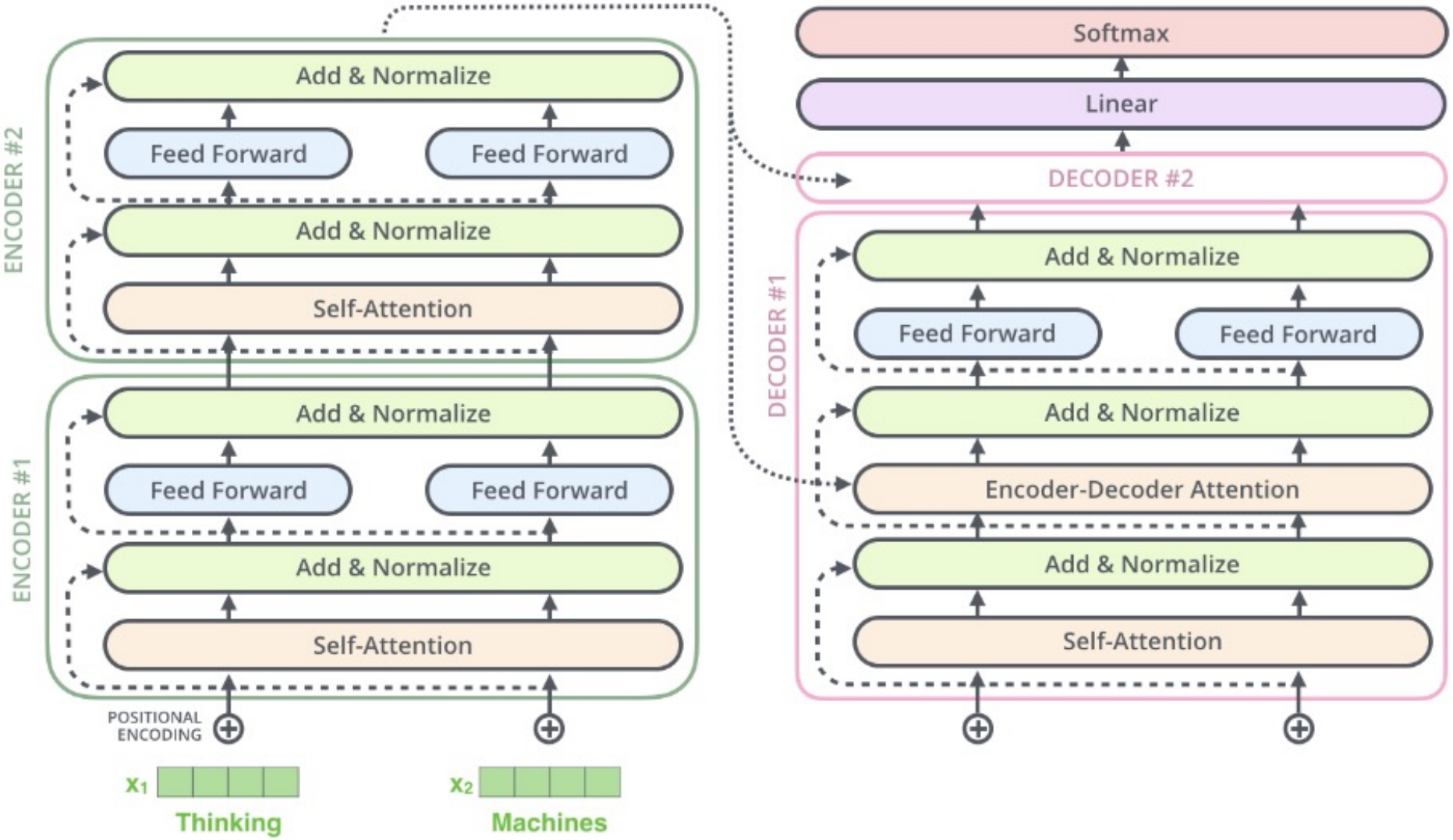
Problem:

No word order information!

Solution:

Add positional embeddings

Transformer block



Transformer advantages

- Number of unparallelizable operations does not increase with sequence length.
- Each "word" interacts with each other

State-of-the-art transformers:

Radford, A., et al. (2018) Open AI	GPT-2: Generative Pre-training for Transformer
Devlin, J., et al. (2018) Google AI Language	BERT:Bidirectional Encoder Representations from Transformers

To do

- Optional reading: **SLP** Ch9
- Review course materials