

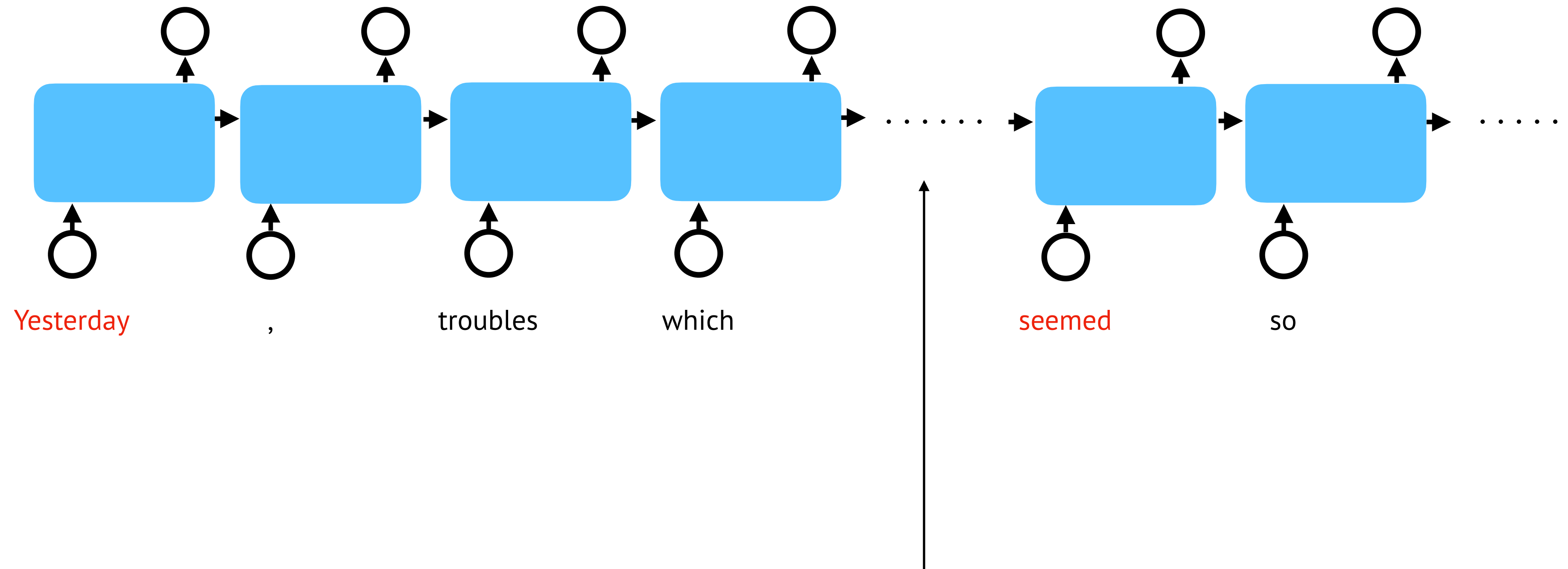
Transformers (1)

COMP3361 — Week 4

Lingpeng Kong

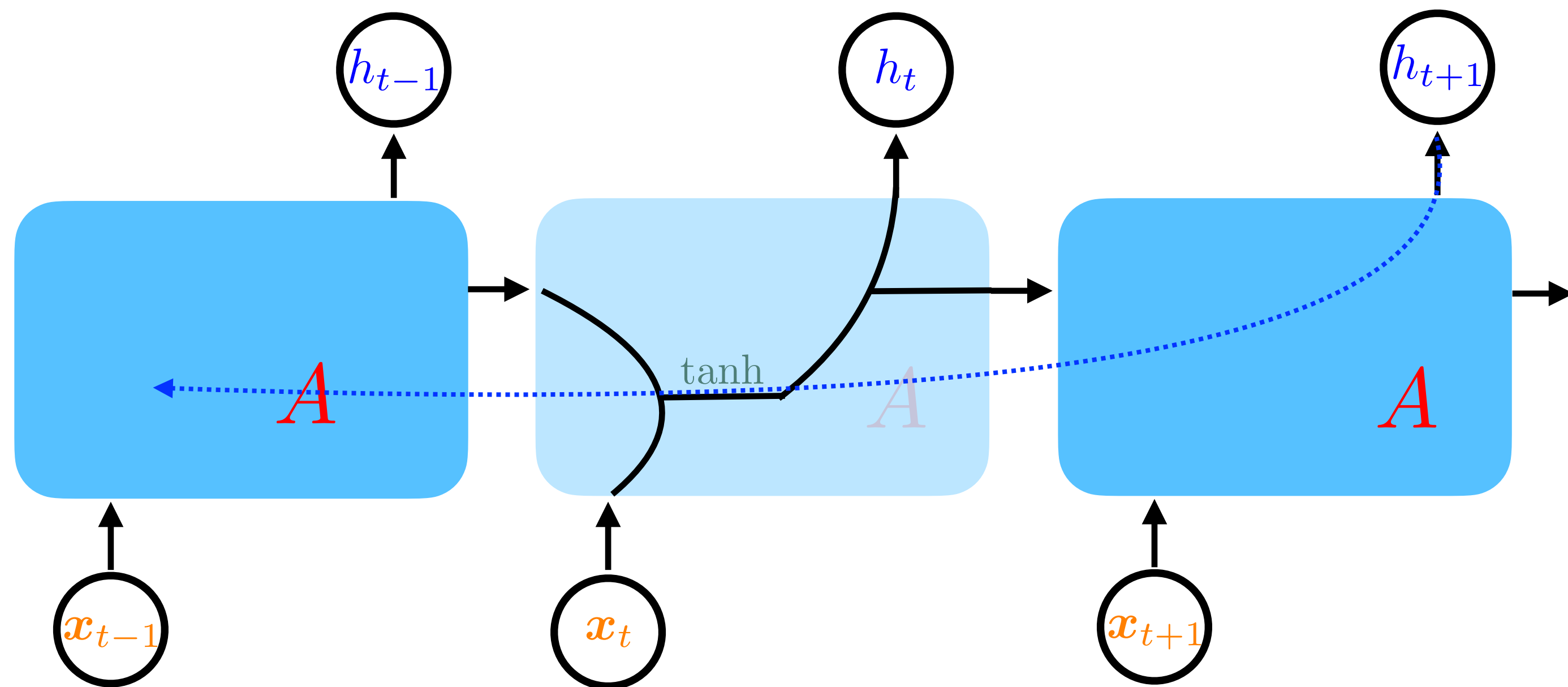
Department of Computer Science, The University of Hong Kong

Recurrent Neural Network



Possibly many steps $[O(N)]$ steps before “yesterday” and “seemed” interact.

Vanishing Gradient in RNNs

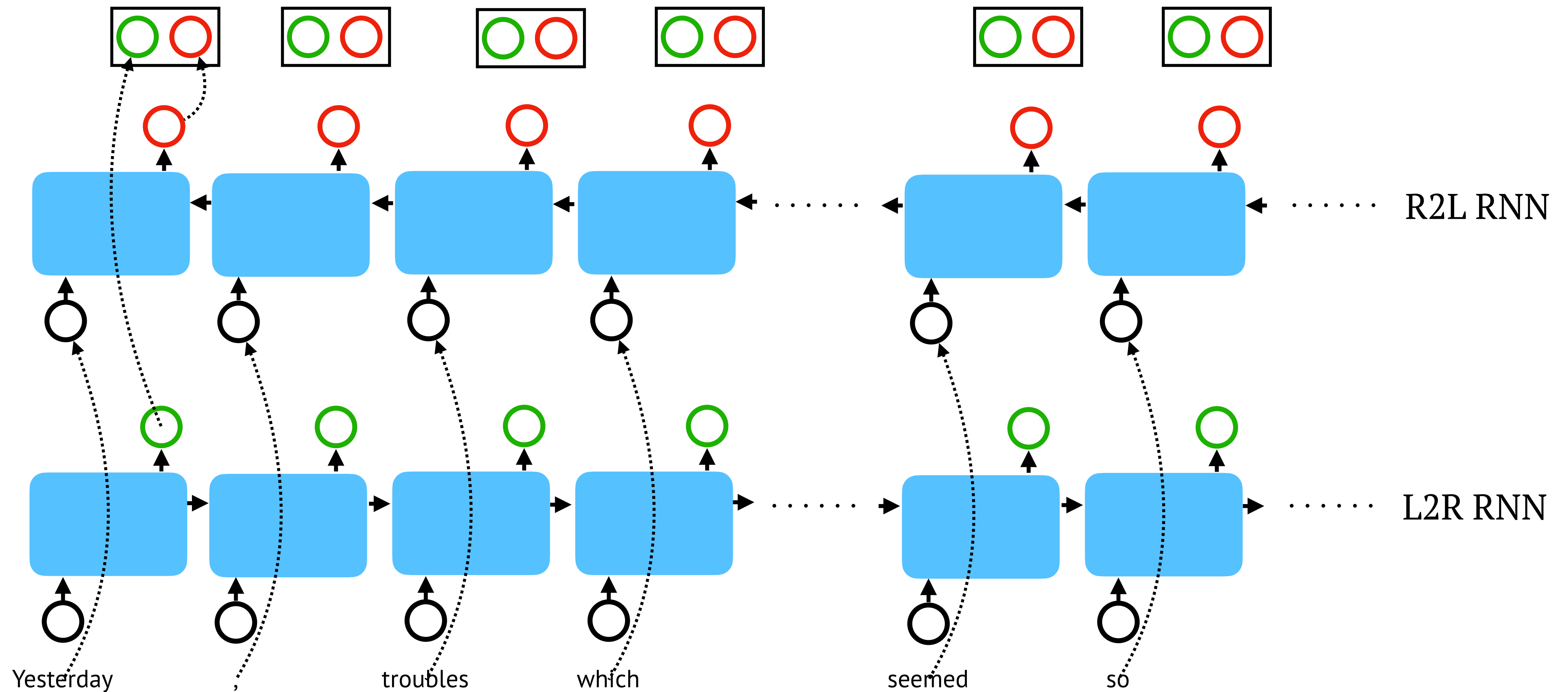


$$h = f(z) \quad \begin{array}{ccc} \textcircled{z} & \xrightarrow{f} & \textcircled{h} \\ \frac{\partial s}{\partial z} = \frac{\partial s}{\partial h} \frac{\partial h}{\partial z} & & \frac{\partial h}{\partial z} \quad \frac{\partial s}{\partial h} \end{array}$$

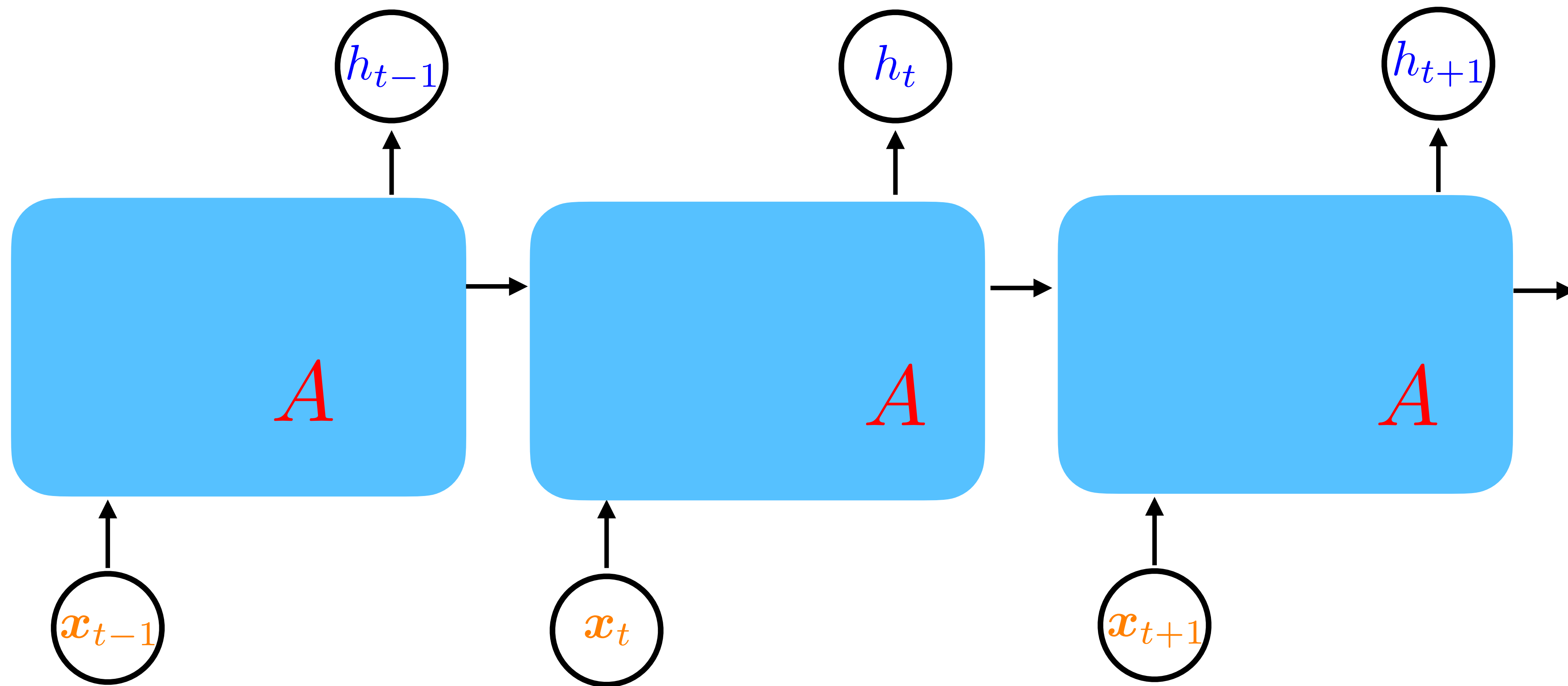
← Gradient Flow Direction

In general, the longer the path, the smaller the gradient signal.

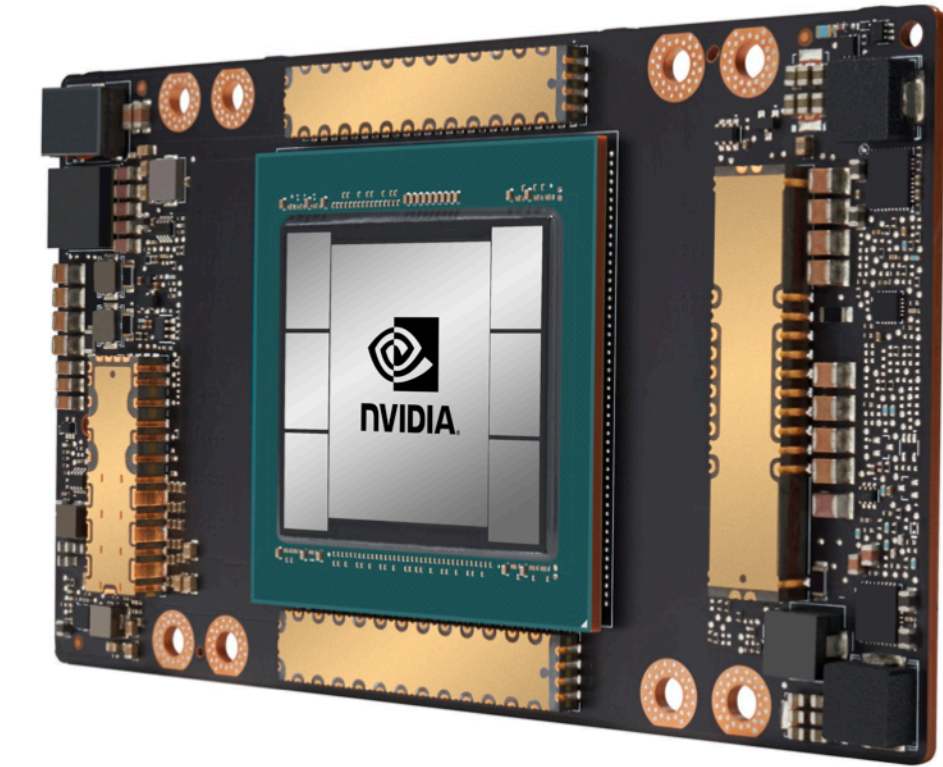
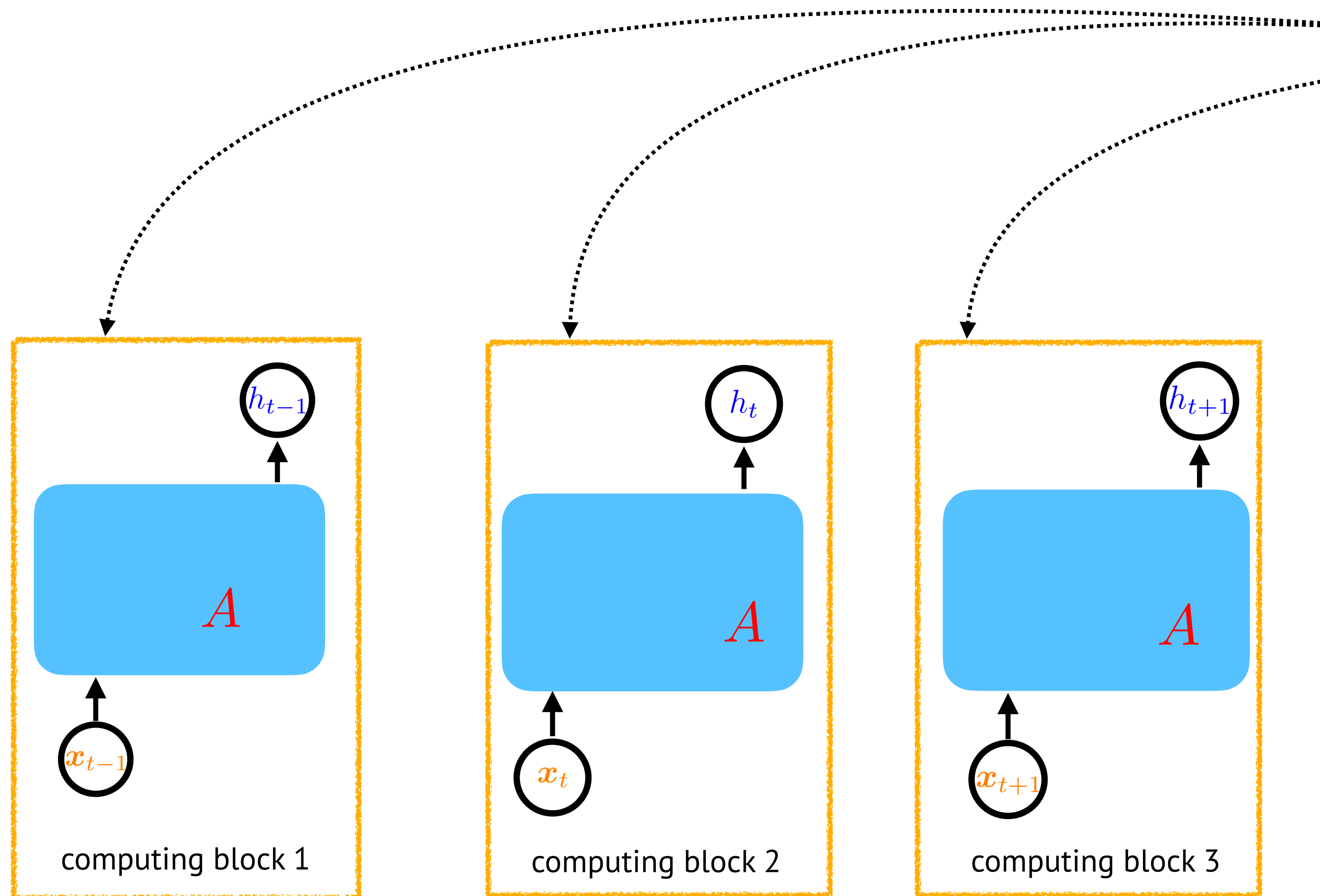
Bidirectional Recurrent Neural Network



Sequential Computation



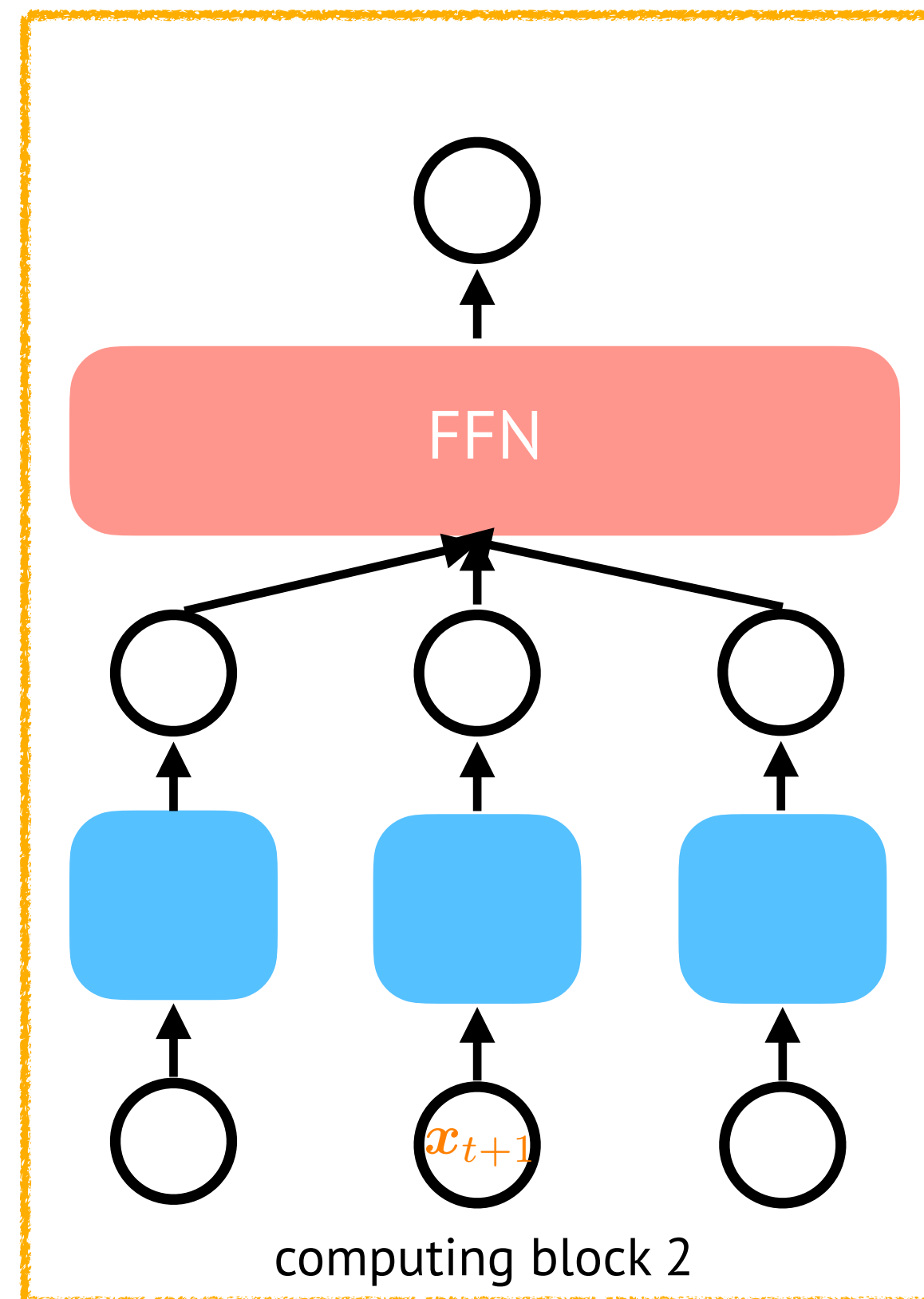
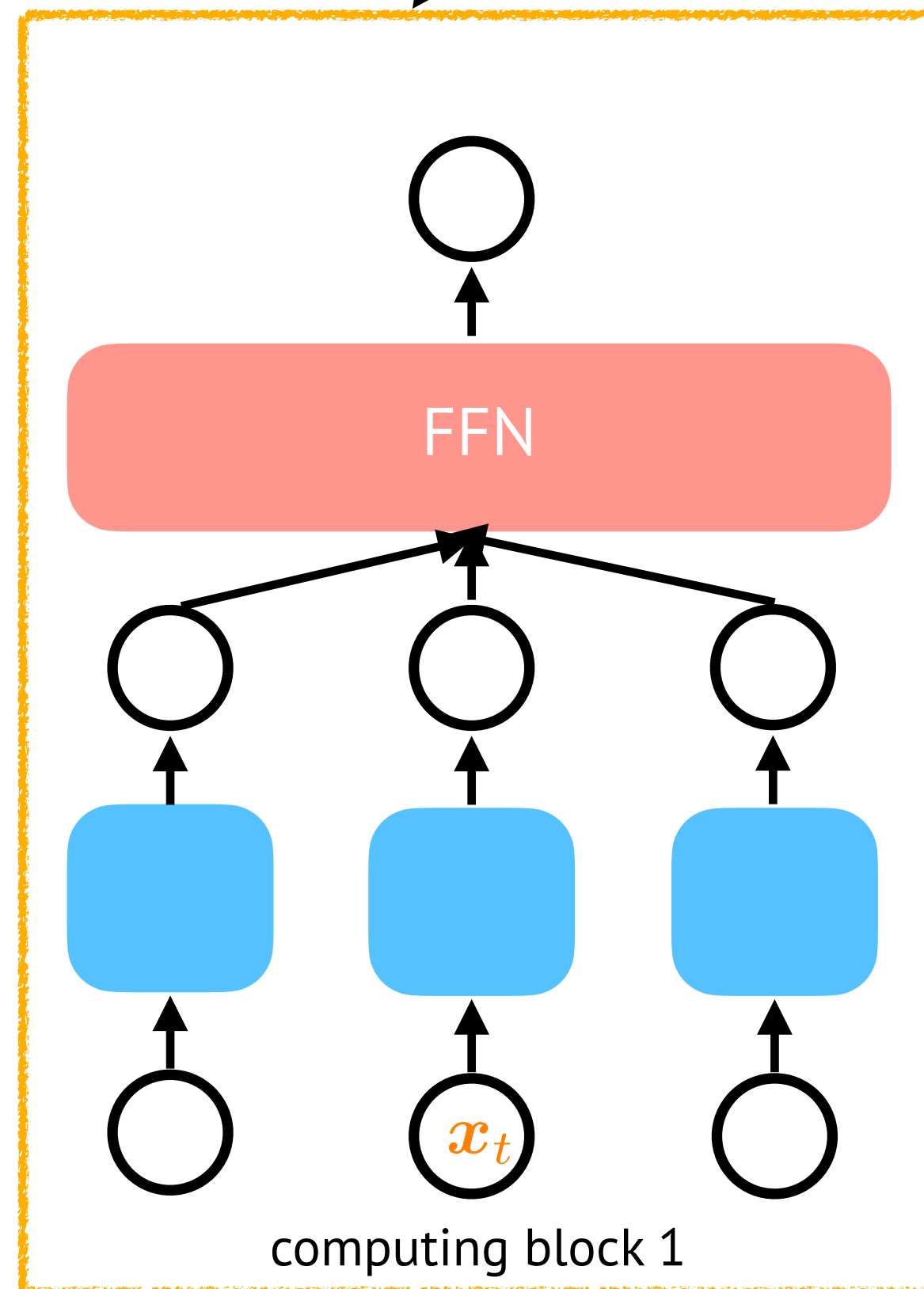
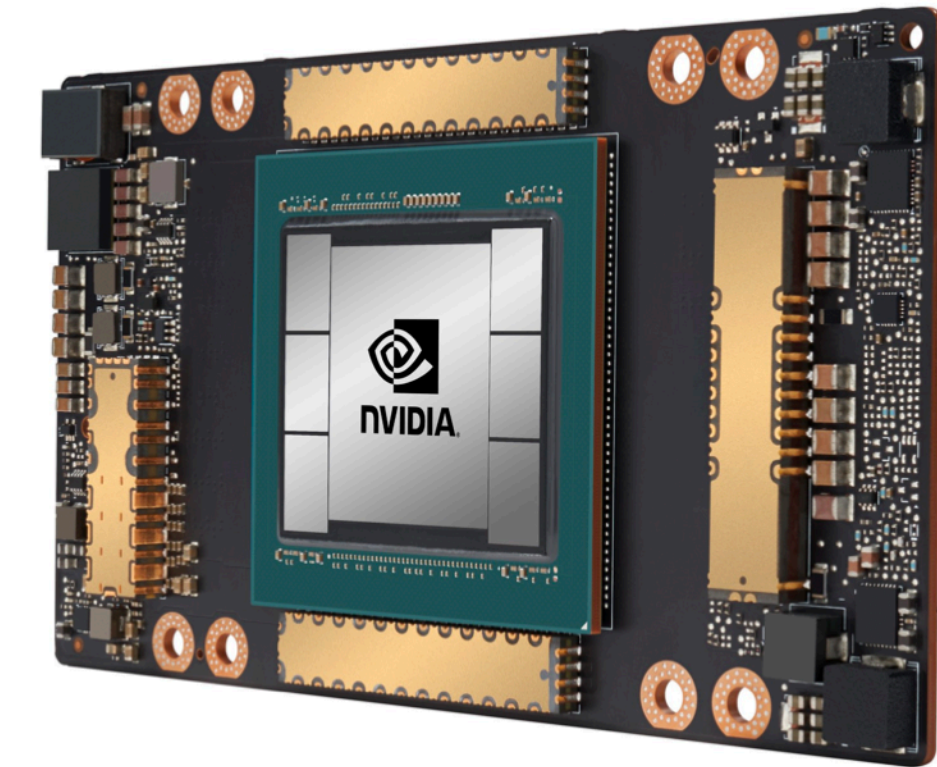
Parallel Computing?



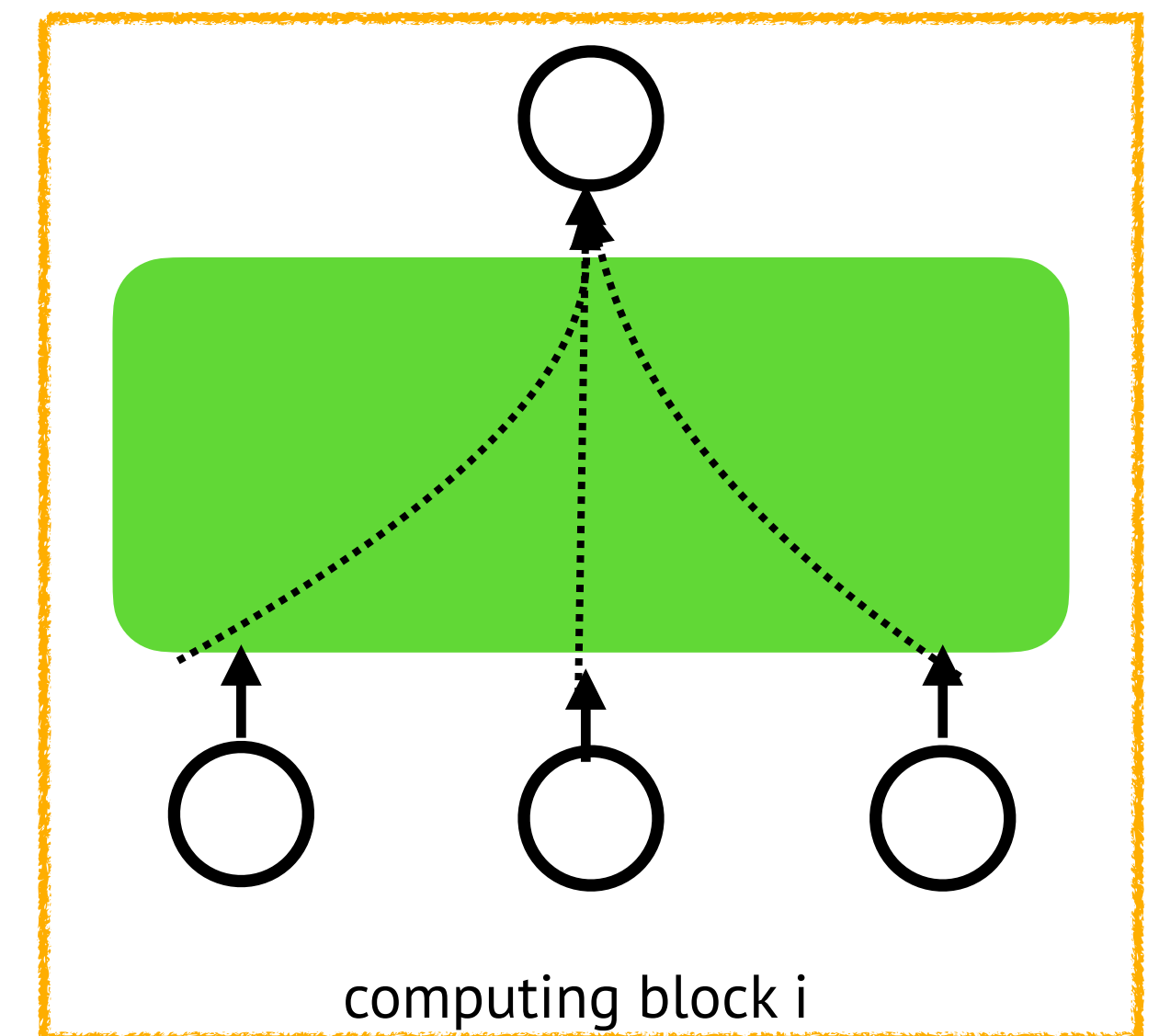
GPU loves parallel computing blocks!

...

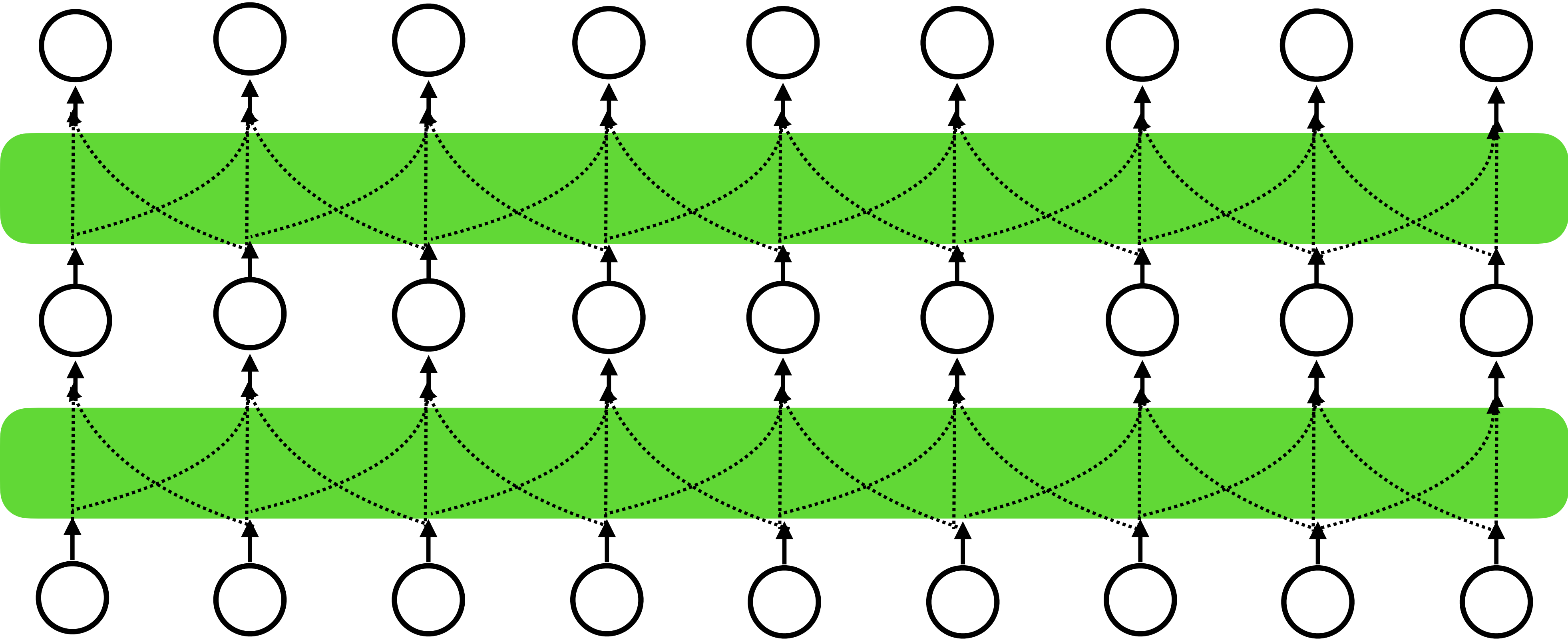
Parallel Computing?



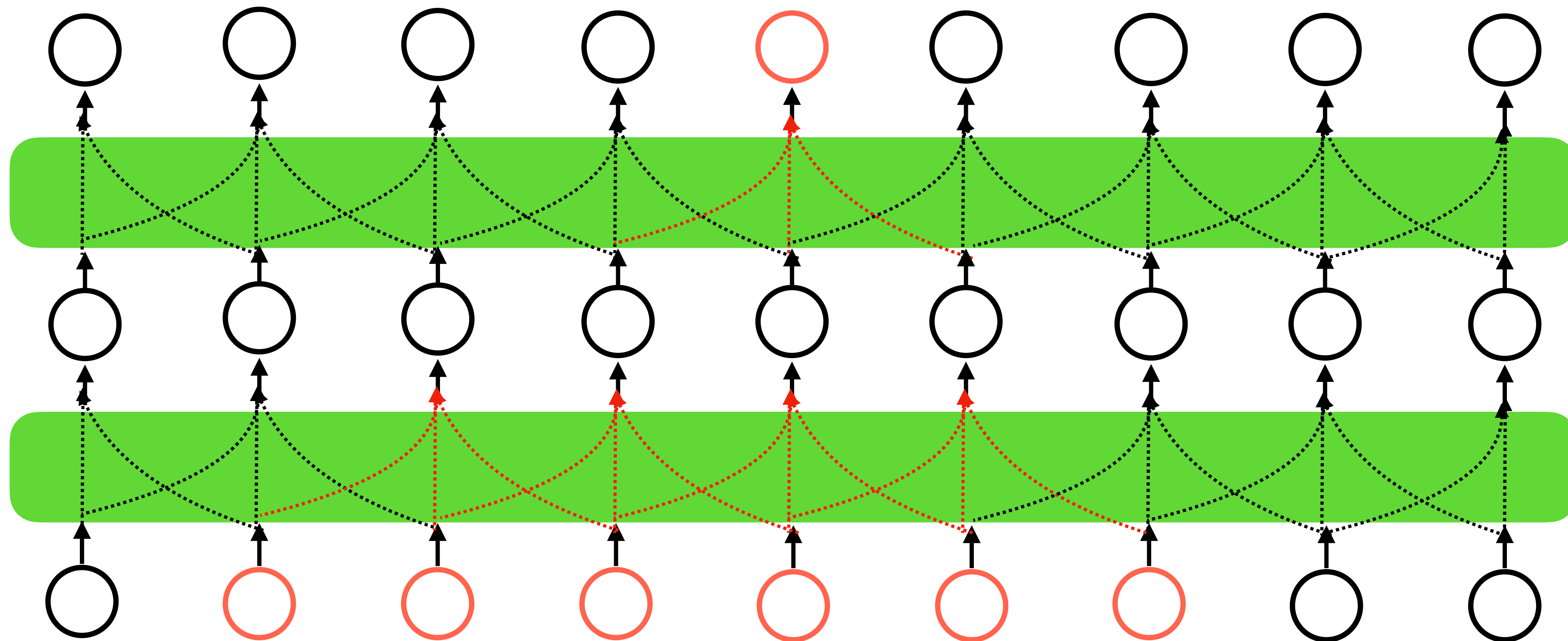
...



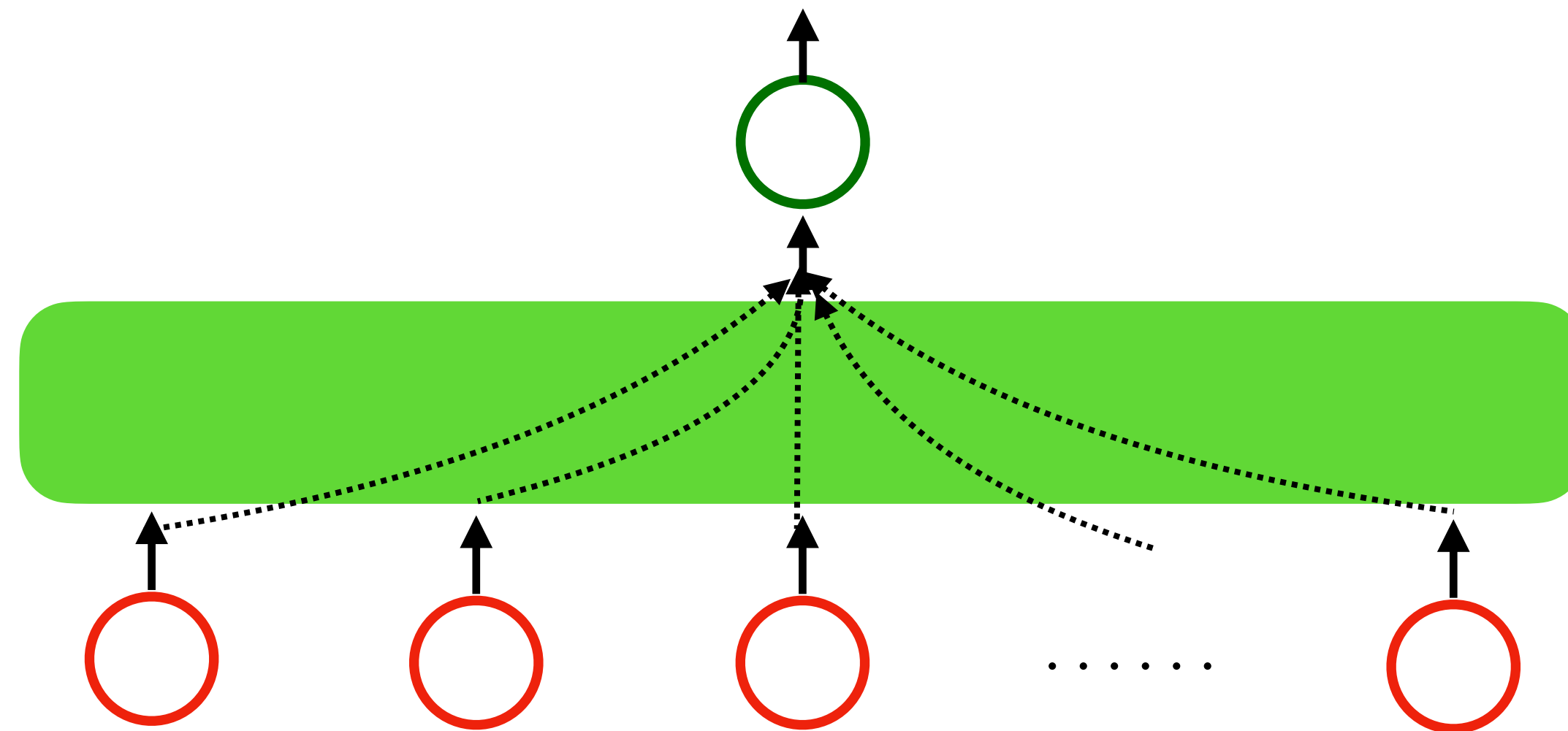
Convolution Style Models



Convolution Style Models

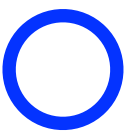
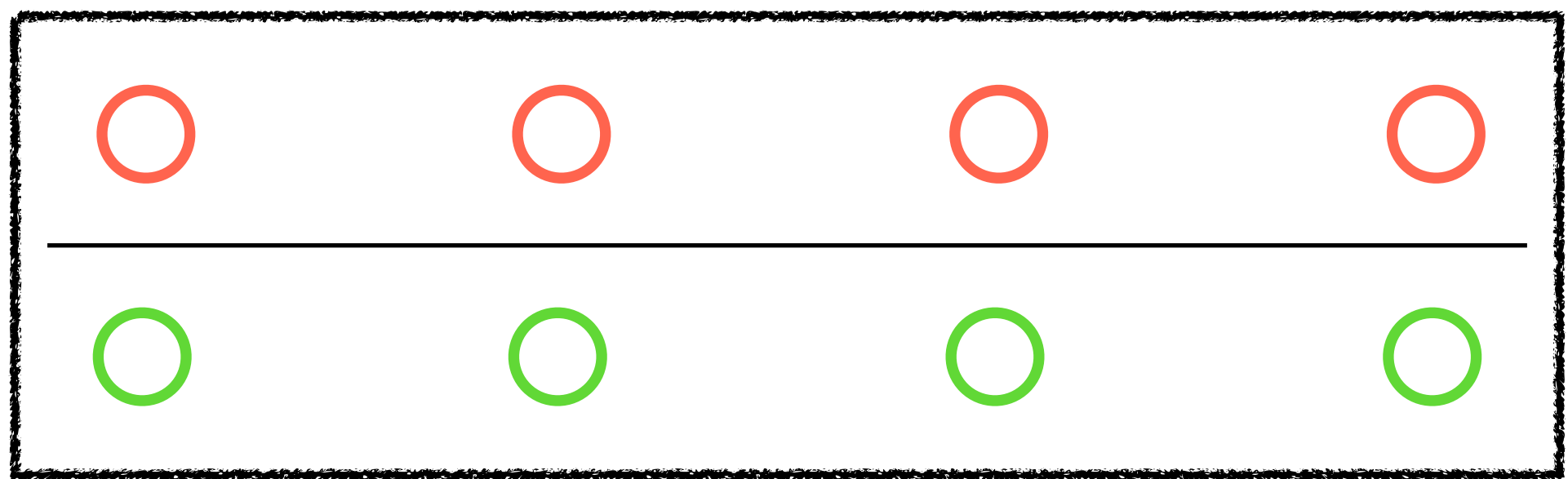


Considering the full sequence as context



How can we achieve this?

Dot-Product-Softmax Attention



Query

Memory (key-value pairs)

Diagram illustrating the Dot-Product-Softmax Attention process:

Query (blue circle) is combined with Memory (key-value pairs) to calculate the attention weights:

Query (blue circle) \cdot k_1 (red circle) $= q \cdot k_1$

Query (blue circle) \cdot k_2 (red circle) $= q \cdot k_2$

Query (blue circle) \cdot k_3 (red circle) $= q \cdot k_3$

Query (blue circle) \cdot k_4 (red circle) $= q \cdot k_4$

These products are passed through a softmax function:

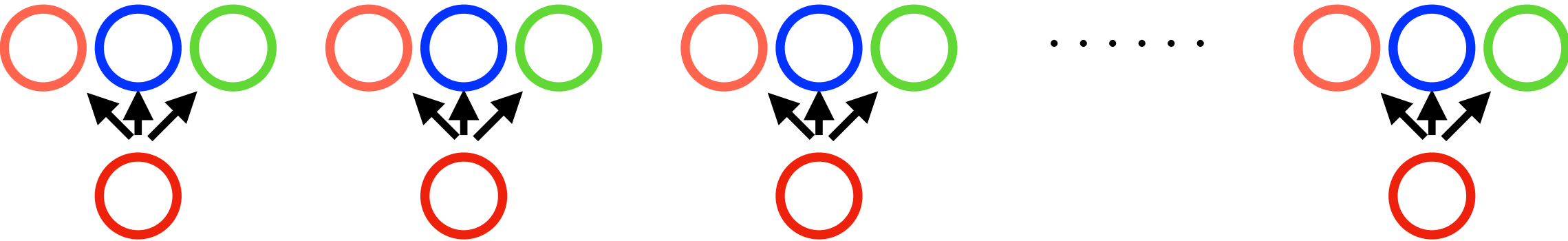
$\text{softmax}(q \cdot k_1, q \cdot k_2, q \cdot k_3, q \cdot k_4) \rightarrow \begin{bmatrix} 0.6 \\ 0.1 \\ 0.2 \\ 0.1 \end{bmatrix}$

The resulting attention weights are then multiplied by the corresponding value vectors (green circles) and summed:

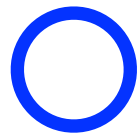
$0.6 \cdot v_1 + 0.1 \cdot v_2 + 0.2 \cdot v_3 + 0.1 \cdot v_4$

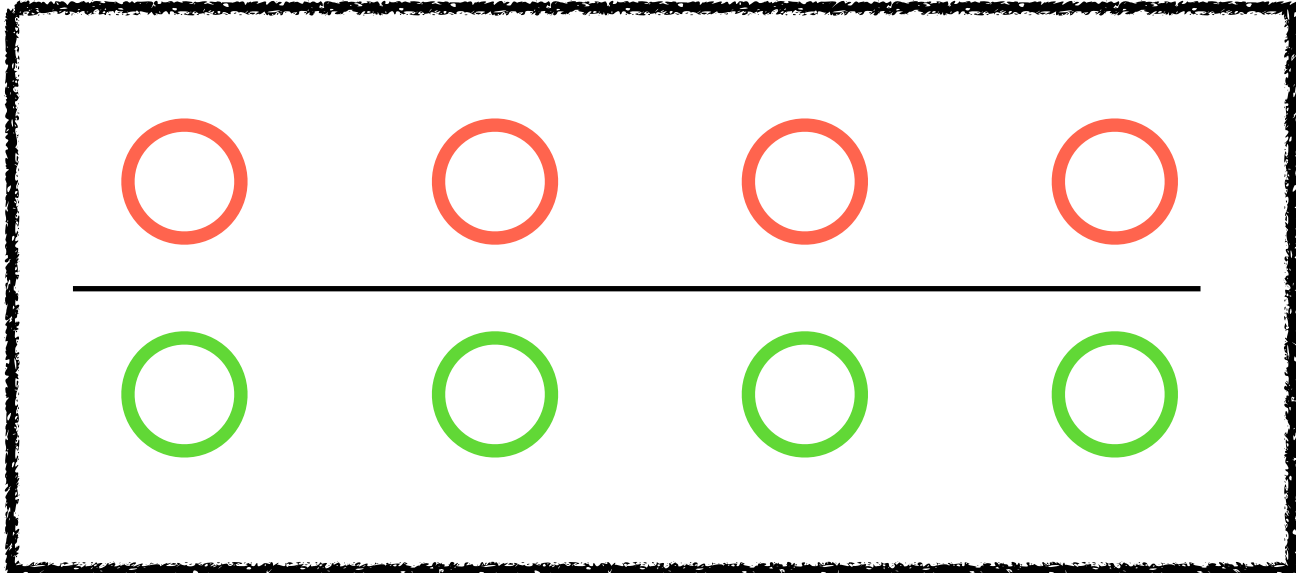
$=$ context vector \mathbf{c} (green circle)

Considering the full sequence as context



Attention Mechanism

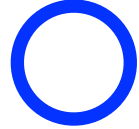

Query



Memory (key-value pairs)

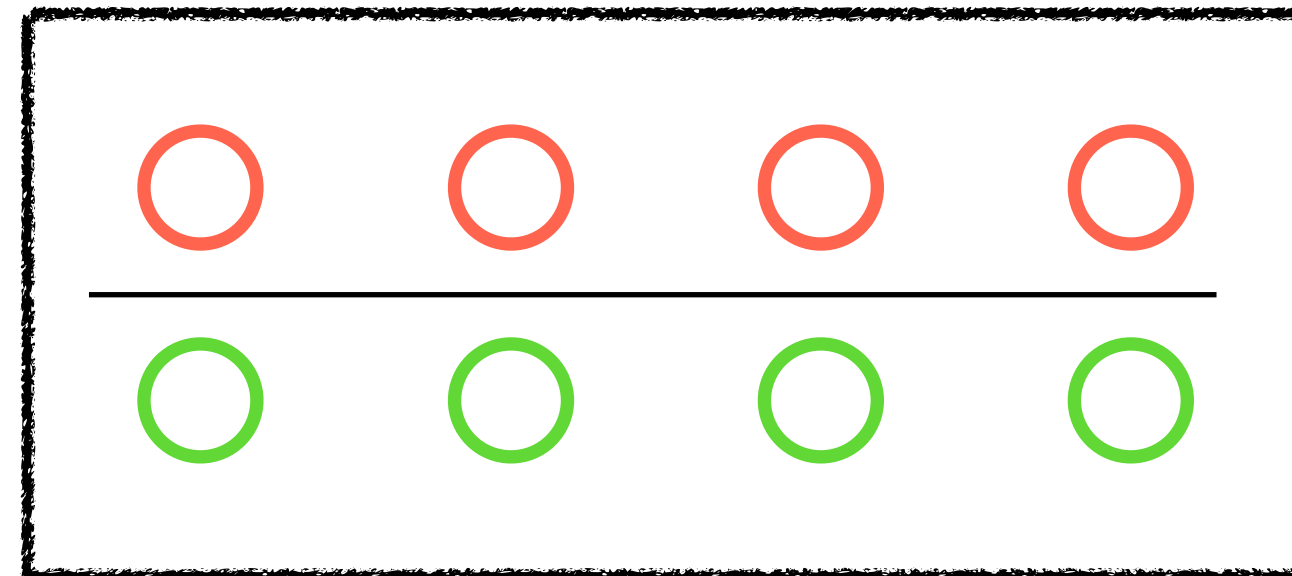


Attention Mechanism


Query

$$0.6 \text{ } \text{○} + 0.1 \text{ } \text{○} + 0.2 \text{ } \text{○} + 0.1 \text{ } \text{○}$$

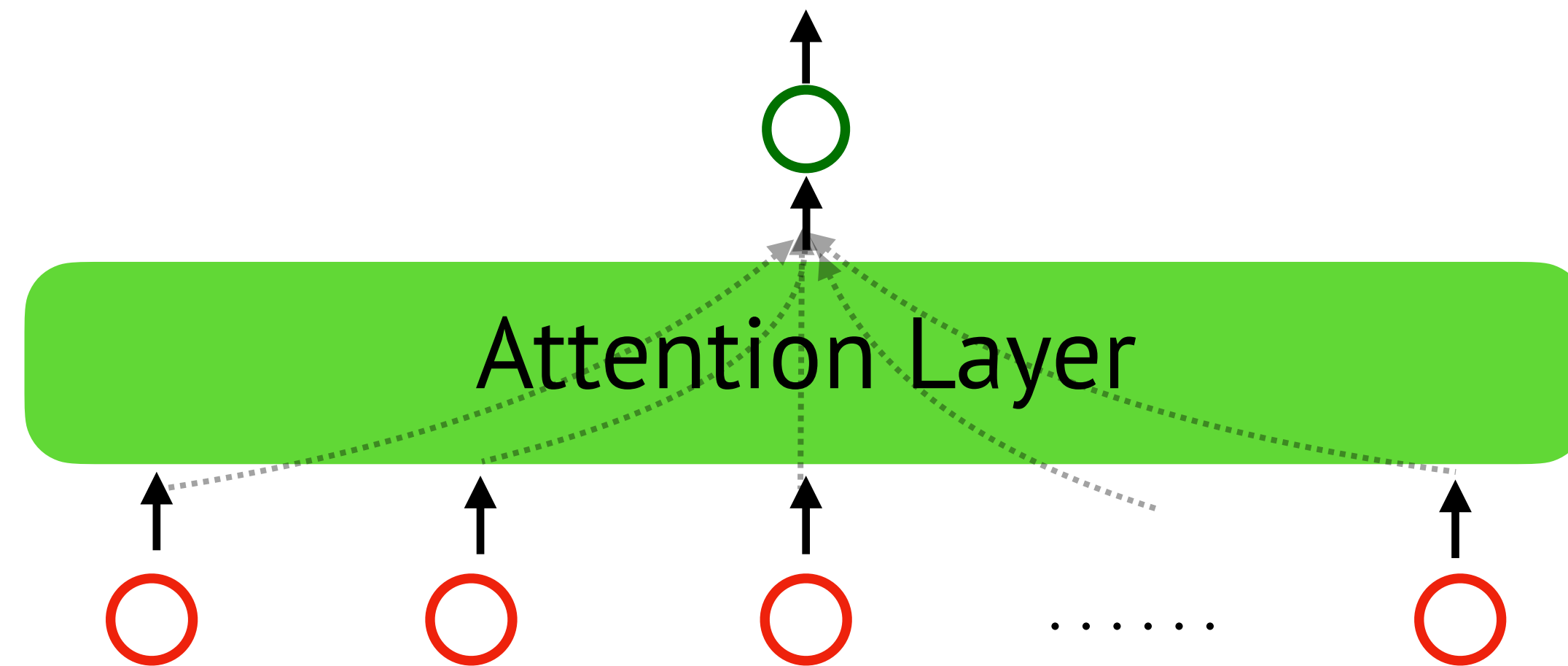
$$= \text{○} \text{ context vector } \mathbf{c}$$



Memory (key-value pairs)

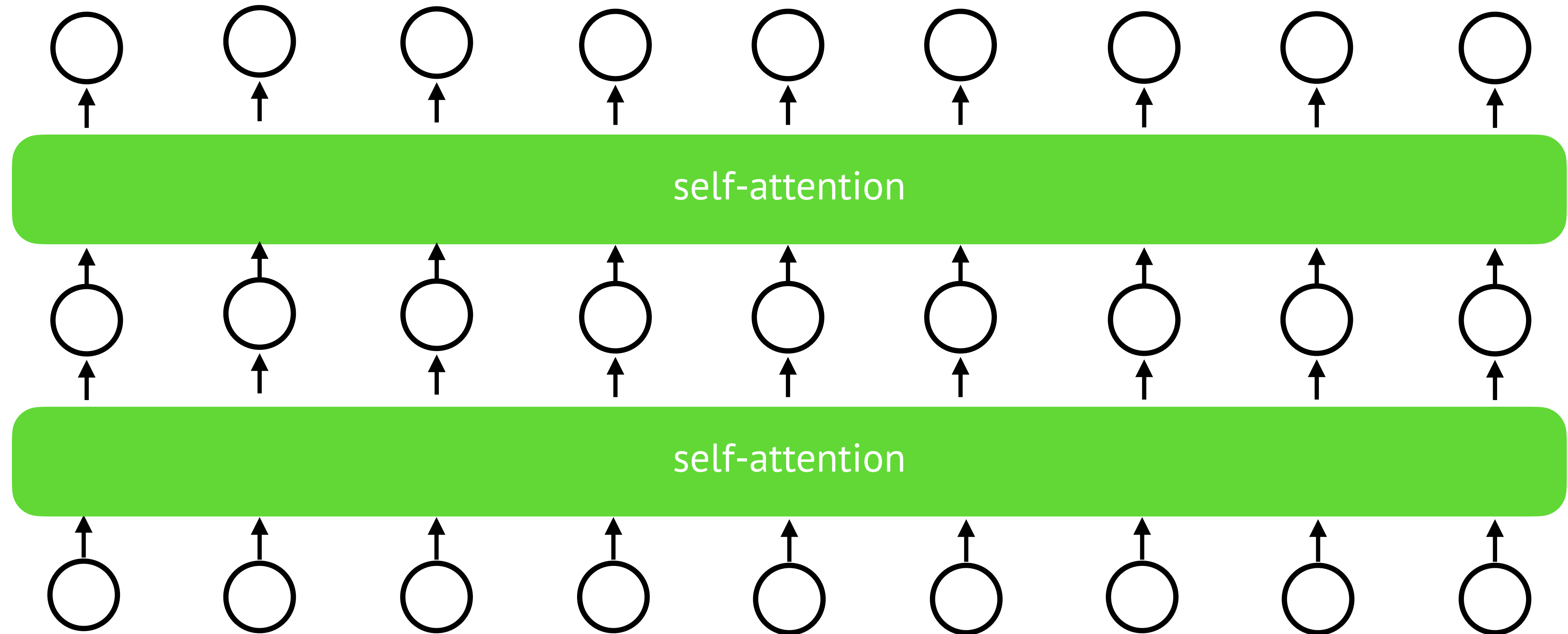


Self-attention

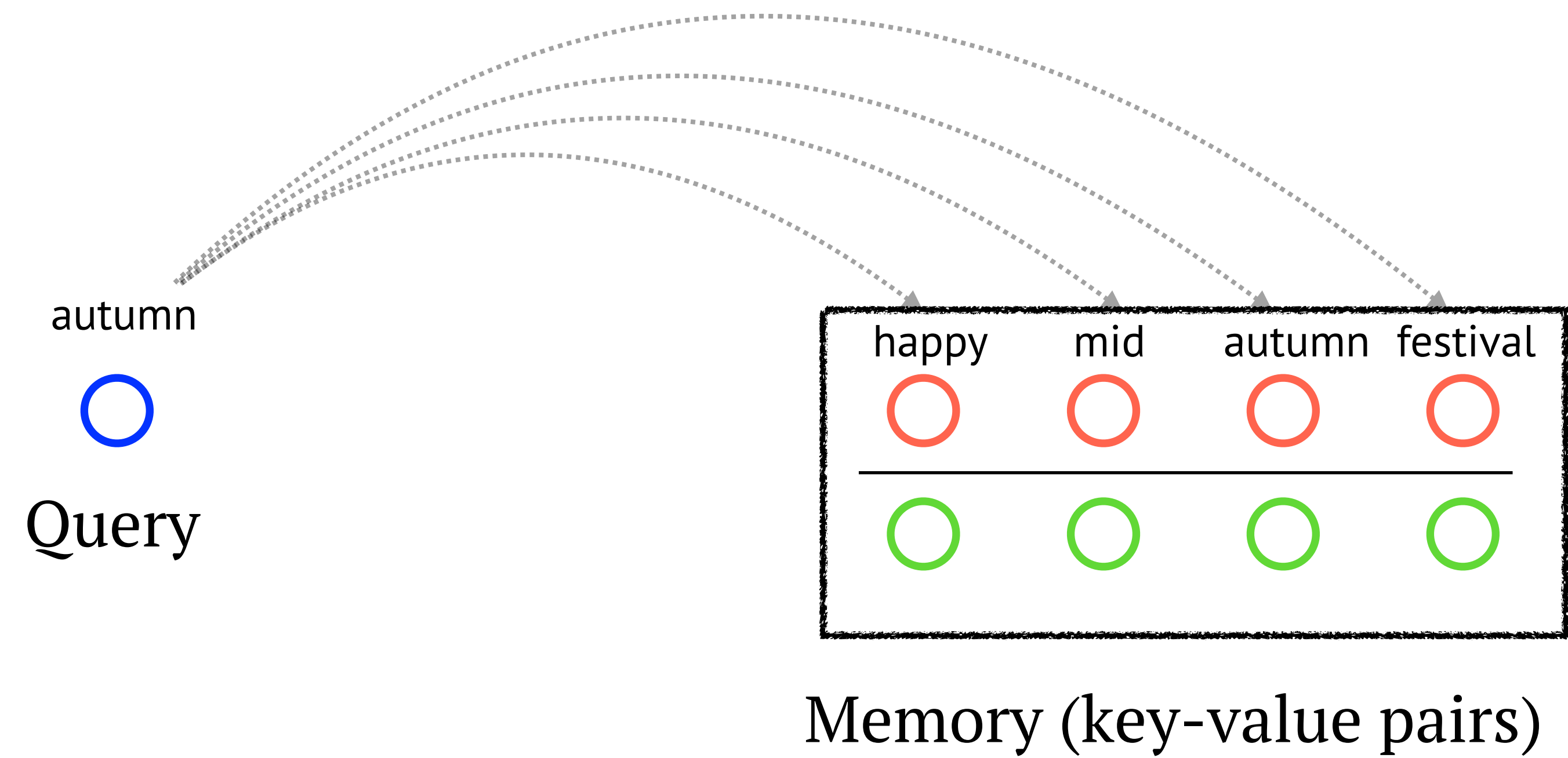


This is almost transformer — except a few things.

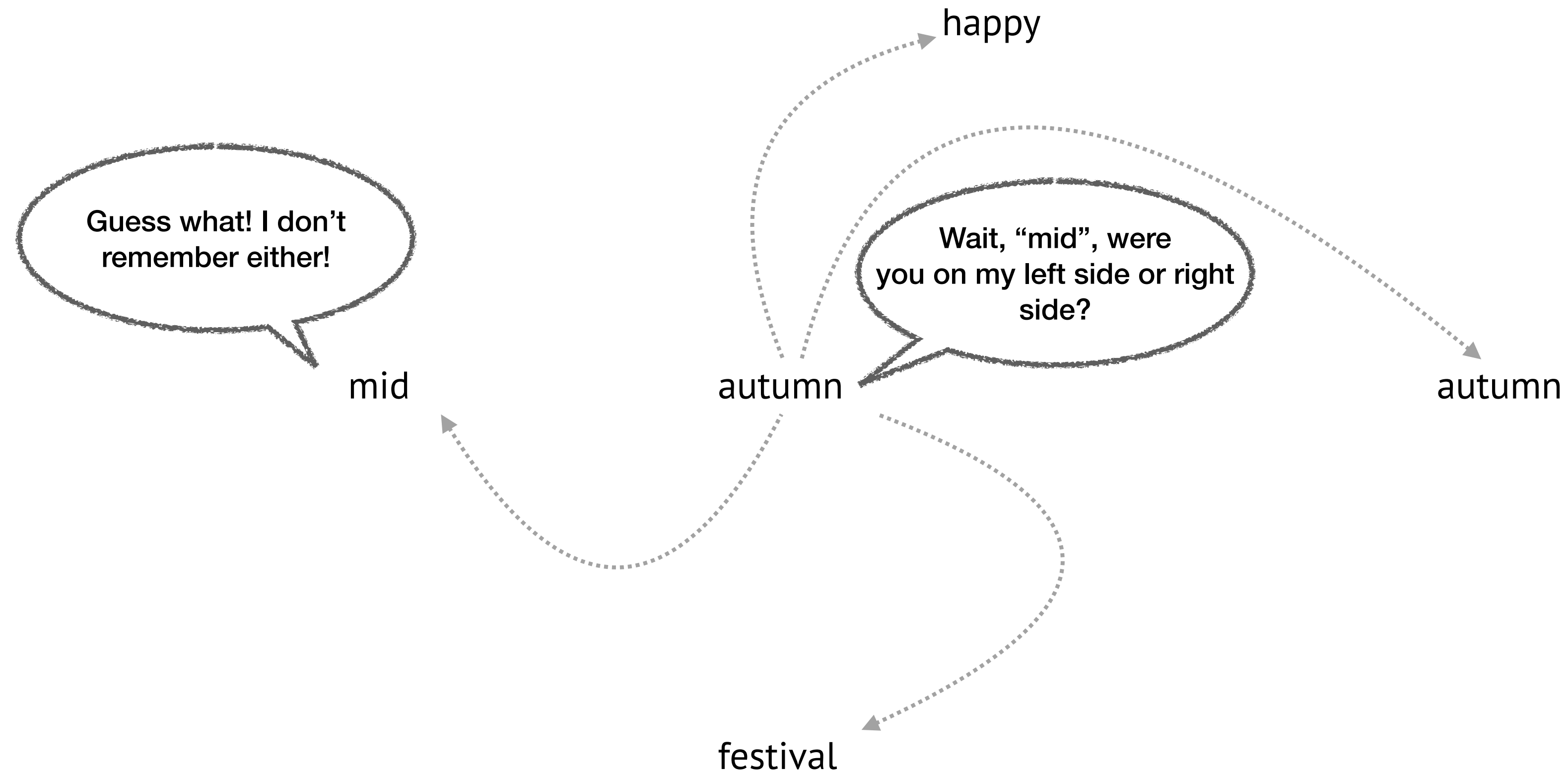
Transformer (almost)



Self-attention in Transformer



Self-attention in Transformer



Positional Embeddings

