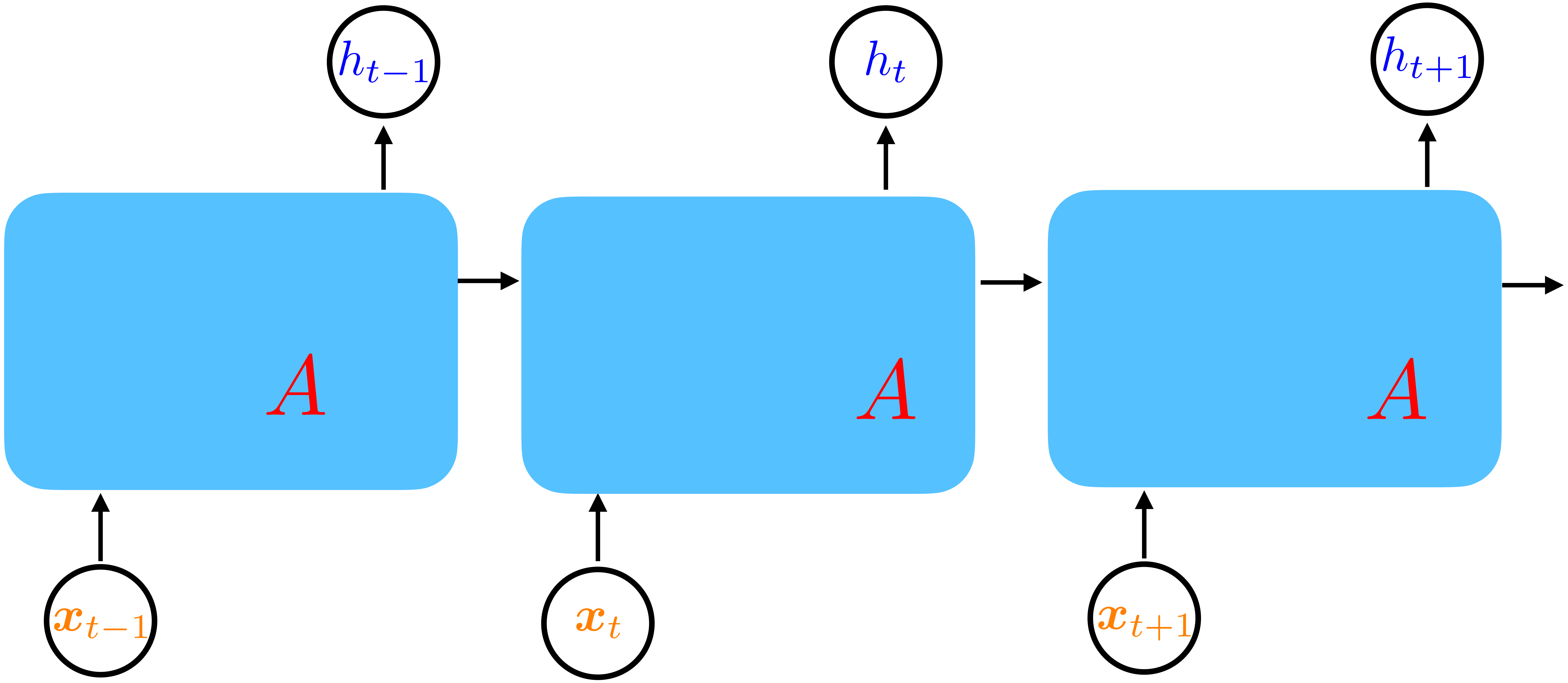
Transformers

COMP3314 — Lecture 9

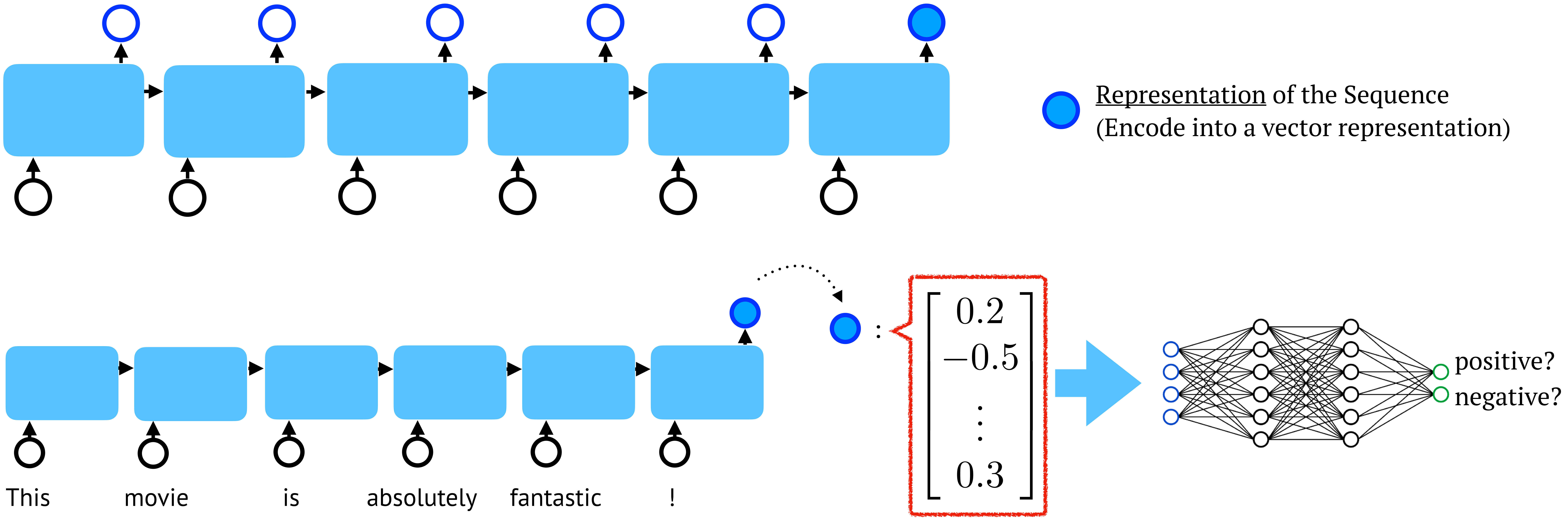
Lingpeng Kong

Department of Computer Science, The University of Hong Kong

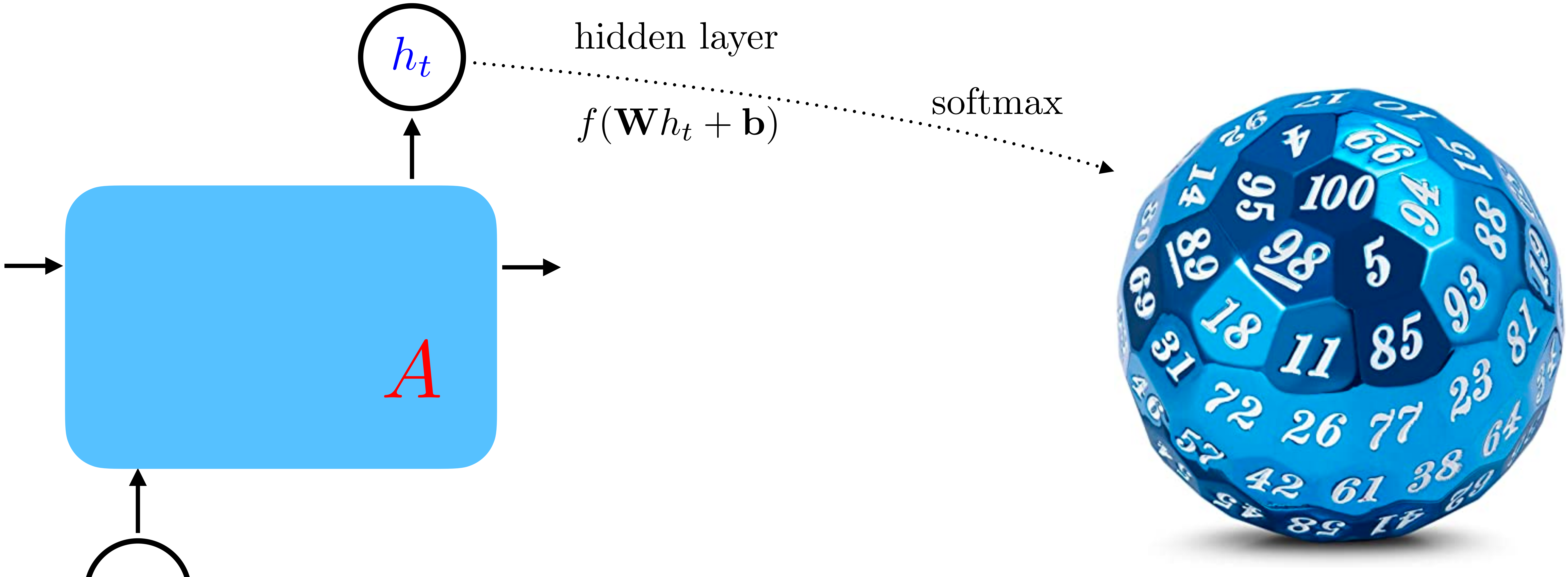
Recurrent Neural Network



RNN as Encoder



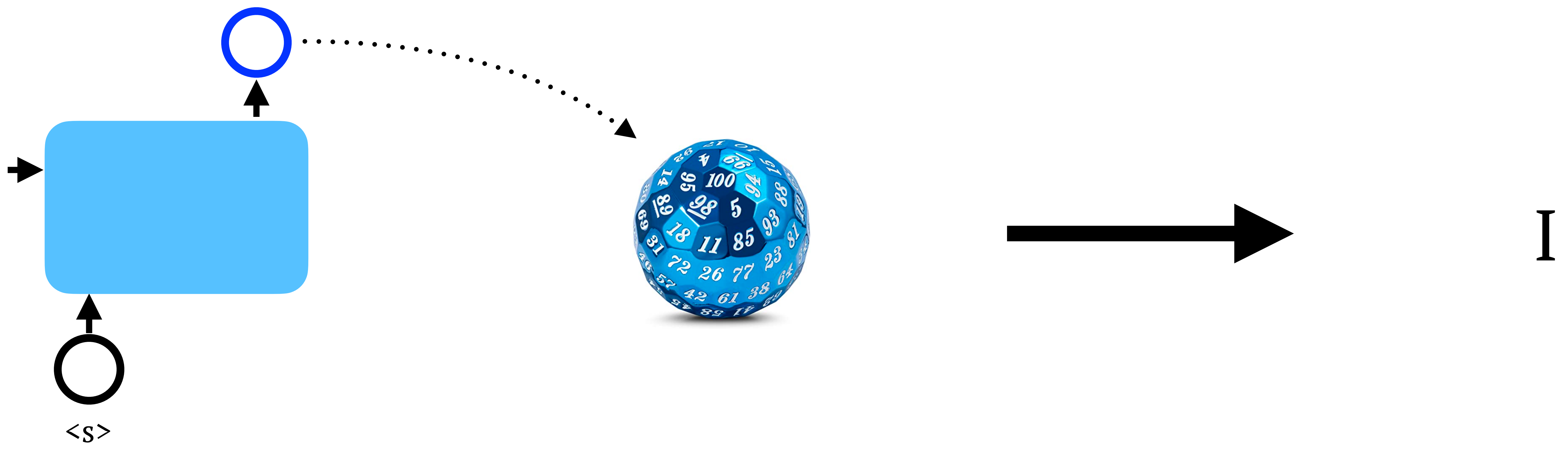
Sample a sentence from RNNLMs



softmax : our neural dice!

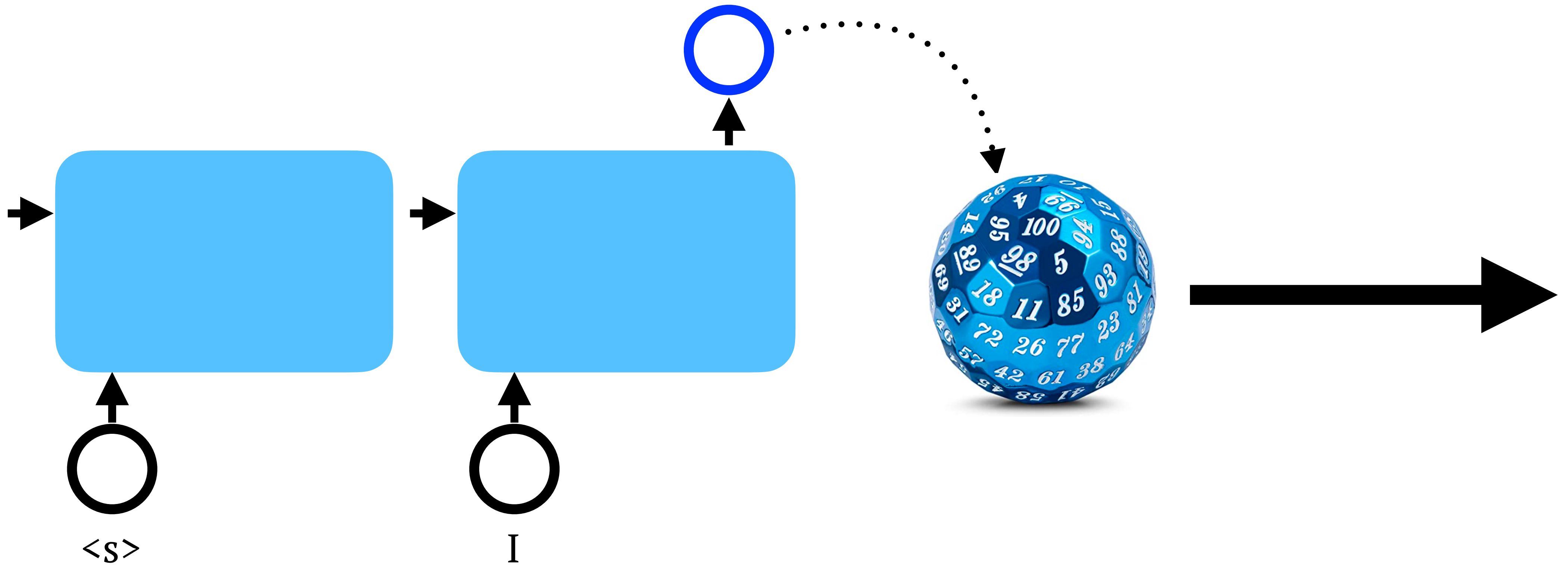
Sample a sentence from RNNLMs

I



Sample a sentence from RNNLMs

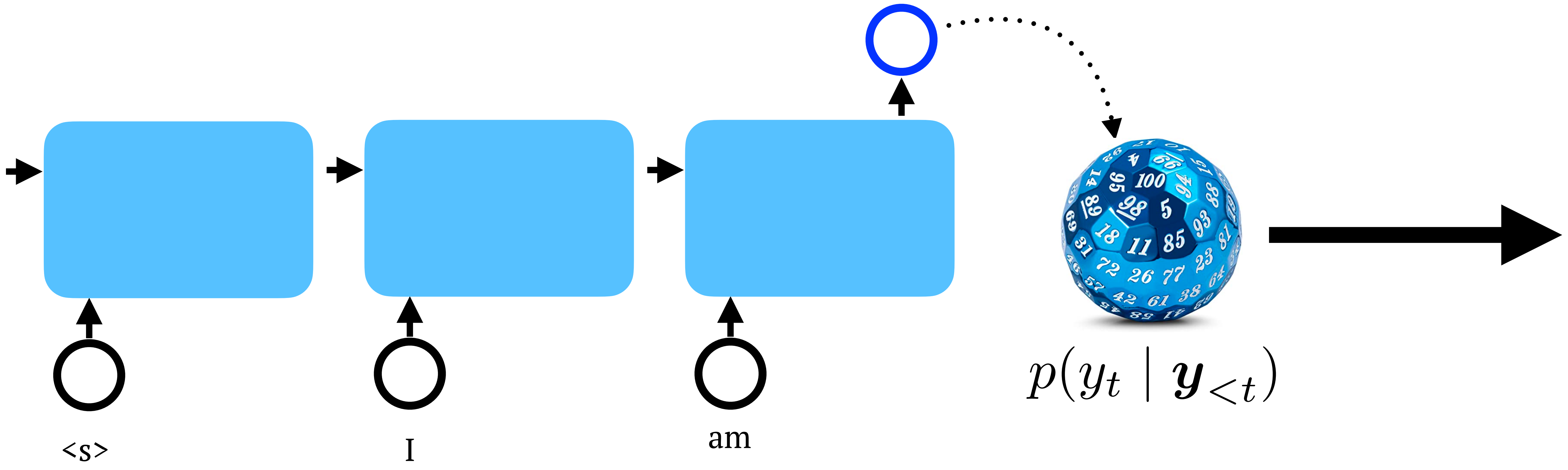
I am



am

Sample a sentence from RNNLMs

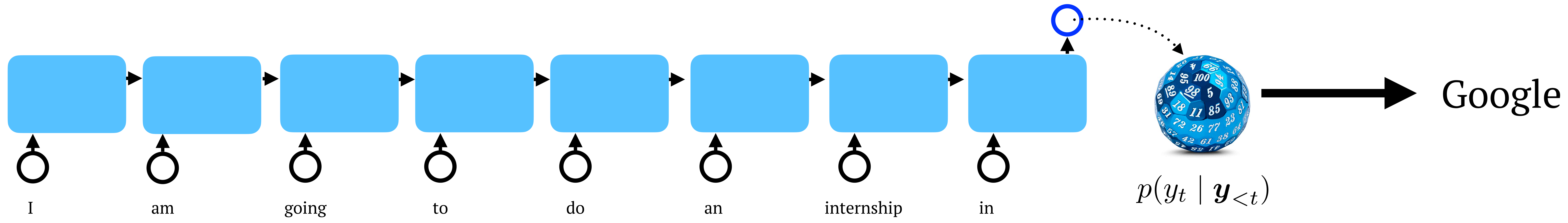
I am going



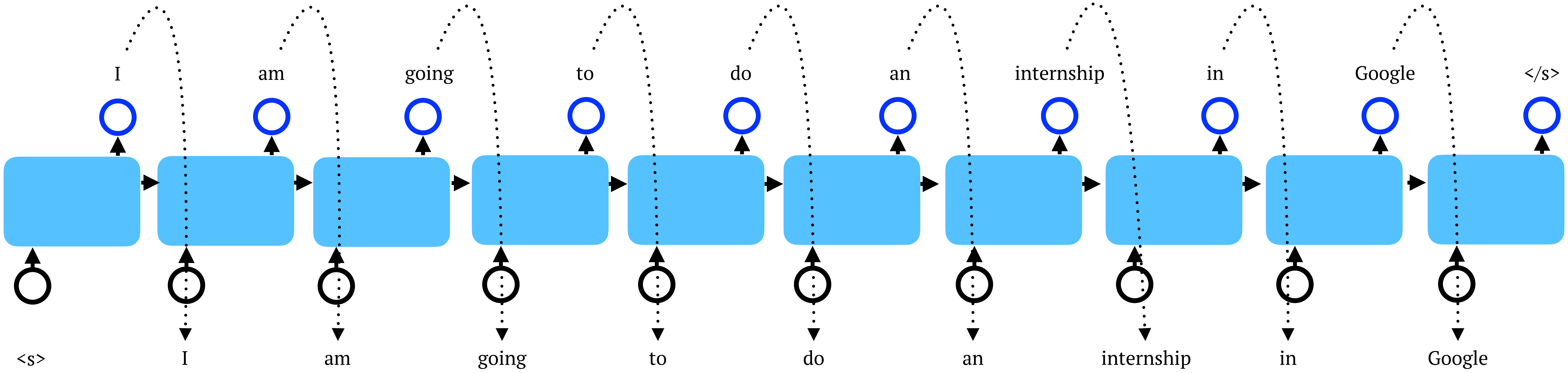
going

Sample a sentence from RNNLMs

I am going to do an internship in Google



RNN as Decoder (RNNLM)



$$p(y_t \mid \boldsymbol{y}_{<t})$$

Machine Translation

中 秋 快 樂 !

$\boldsymbol{x}$

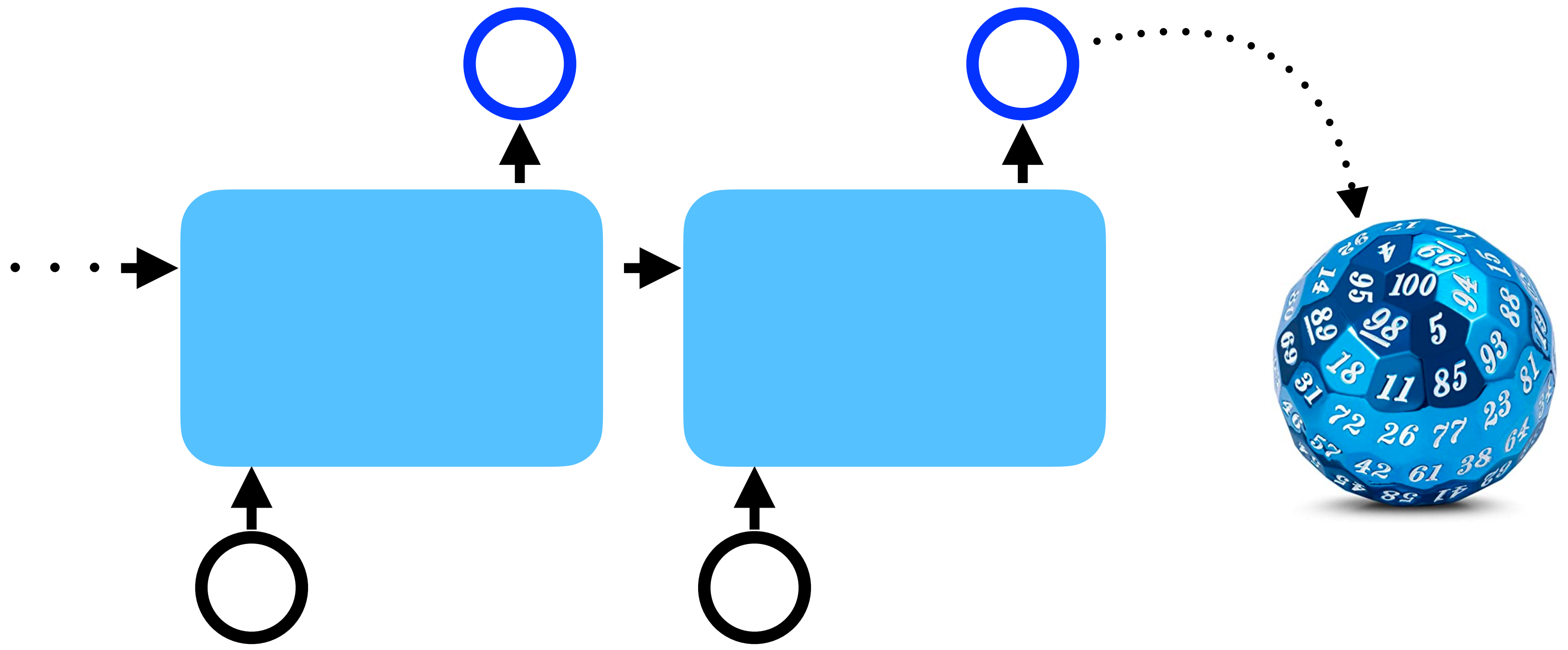
Happy mid autumn festival !

$\boldsymbol{y}$

Happy mid autumn festival !

$$p(\boldsymbol{y}) = p(y_1 \ldots y_n) = \prod_{t=1}^{n} p(y_t \mid \boldsymbol{y}_{<t})$$

$p(y_t \mid \boldsymbol{y}_{<t})$



Machine Translation

中 秋 快 樂 !

$\boldsymbol{x}$

Happy mid autumn festival !

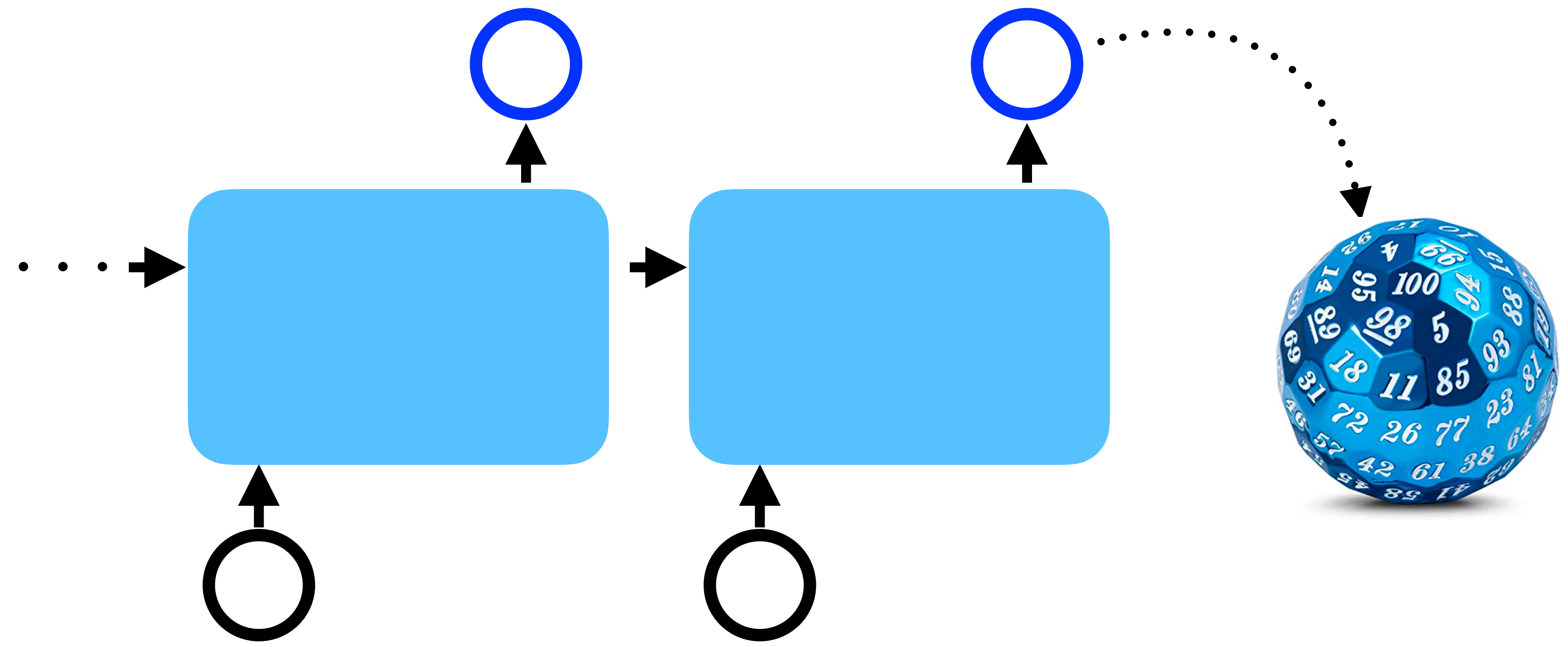
$\boldsymbol{y}$

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = p(y_1 \ldots y_n \mid x_1 \ldots x_m) = \prod_{t=1}^{n} p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x})$$

target source

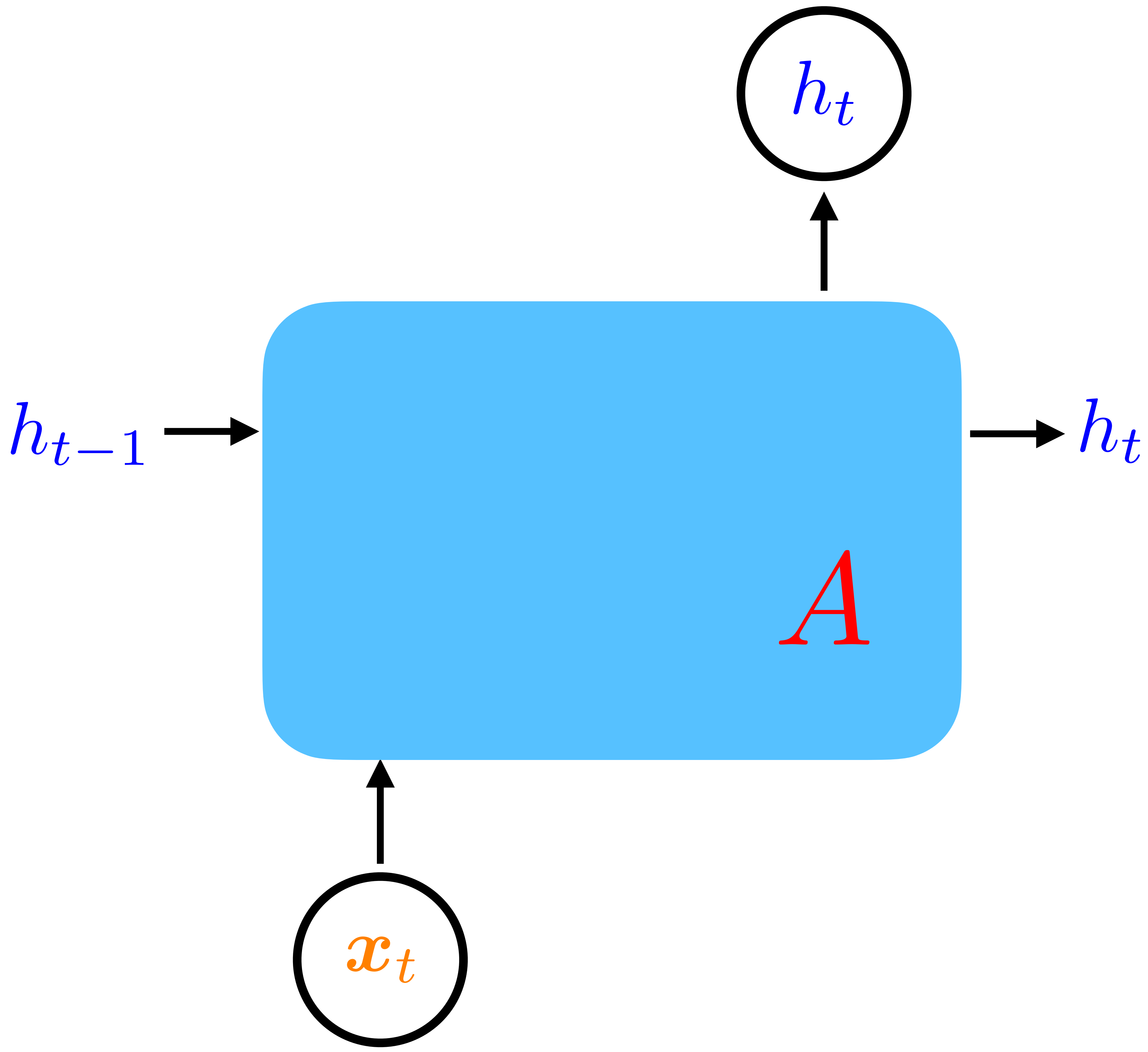
Conditional Language Model

$p(y_t \mid \boldsymbol{y}_{<t})$



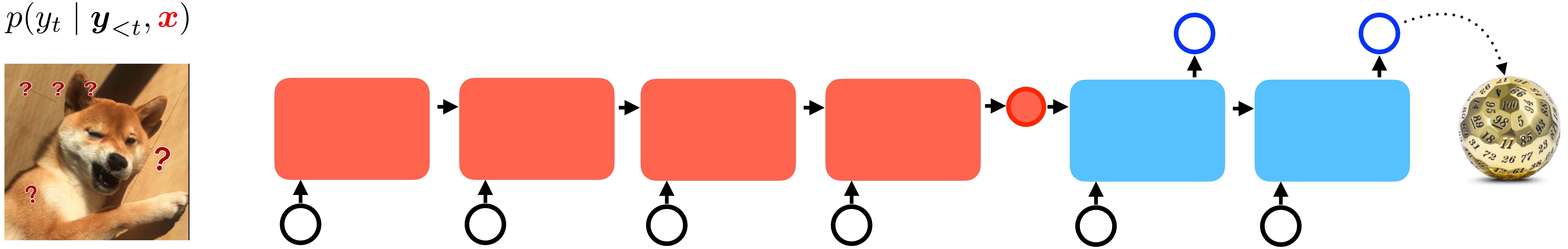
$p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x})$

Recurrent Neural Network



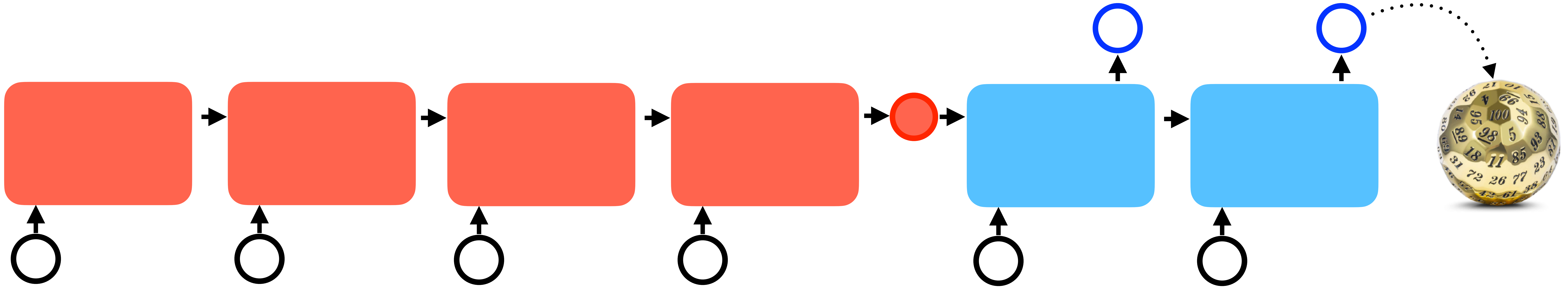
Encoder + Decoder

$p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x})$



Sequence to Sequence Model

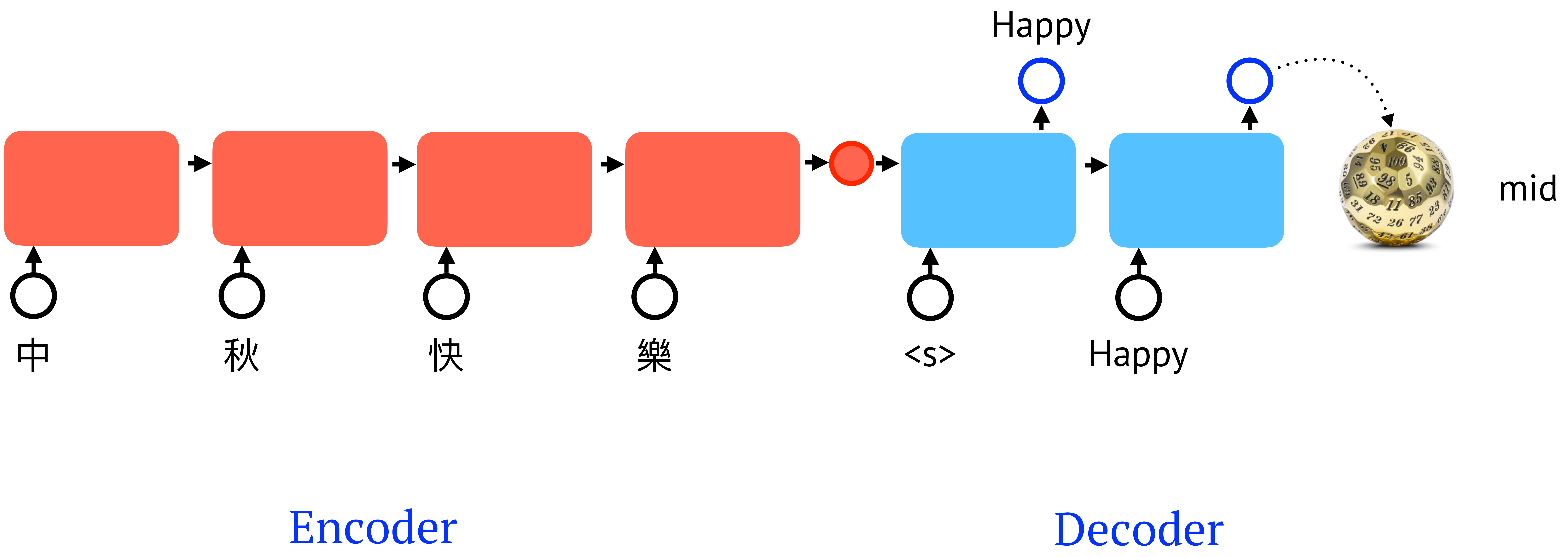
$p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x})$



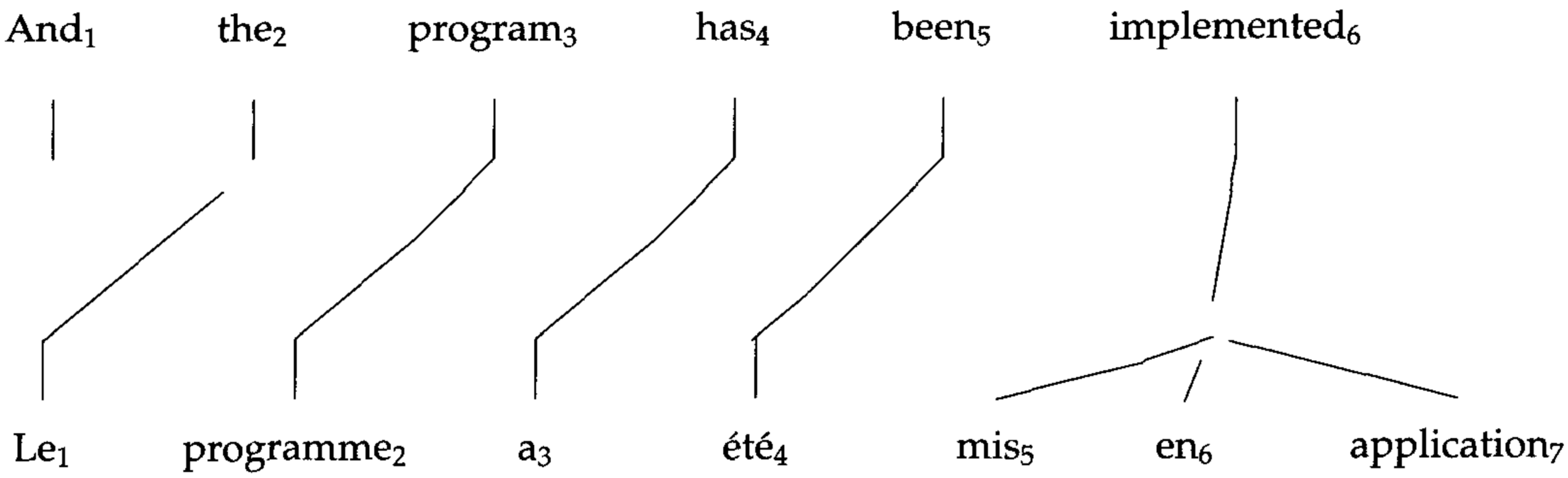
Encoder

Decoder

Sequence to Sequence Model



Alignment in Machine Translation

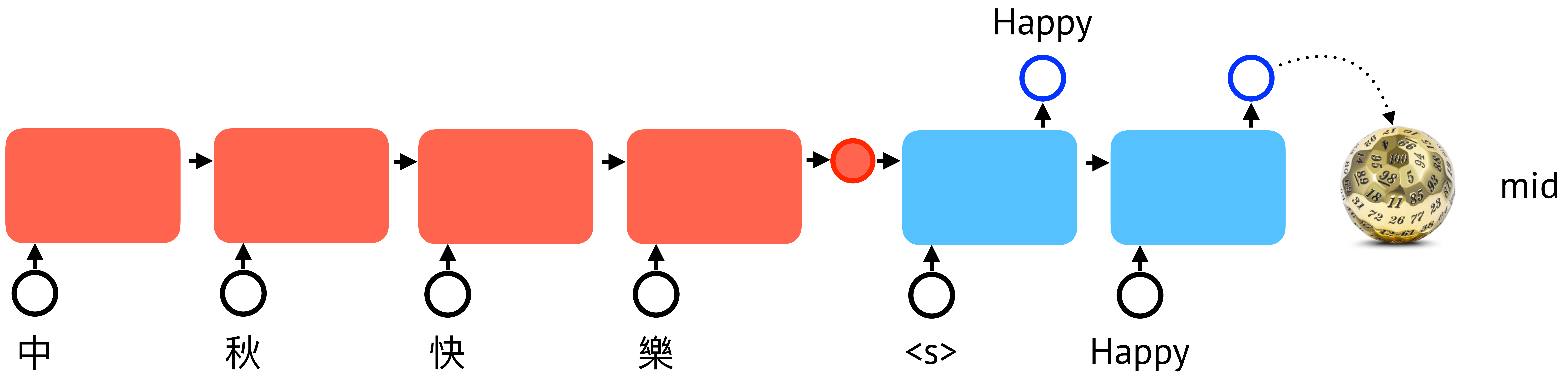


Some words might have no “counter-part”.

Alignment can be many-to-one (or one-to-many).

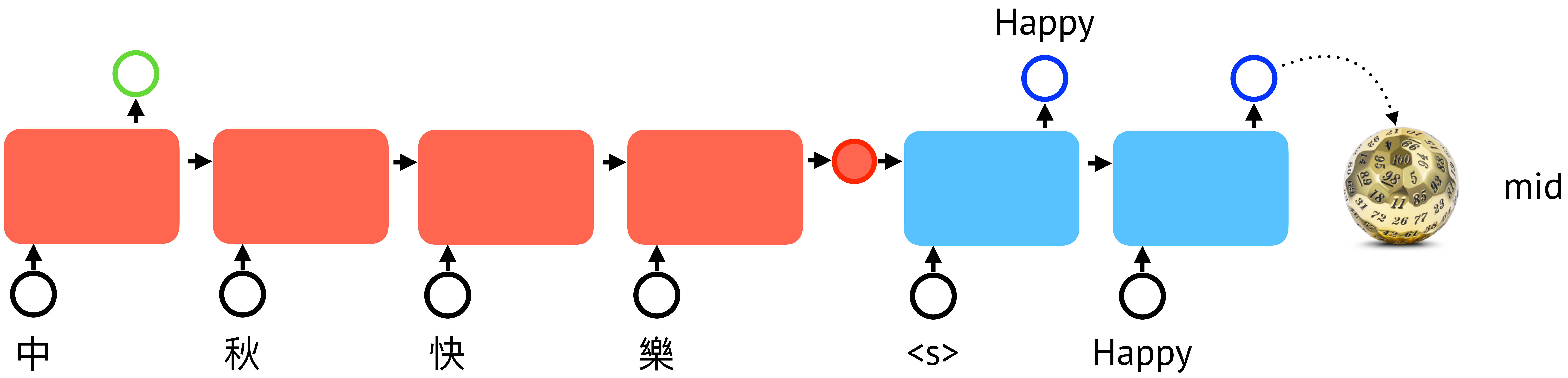
(Brown et al, 1993)

Sequence to Sequence Model

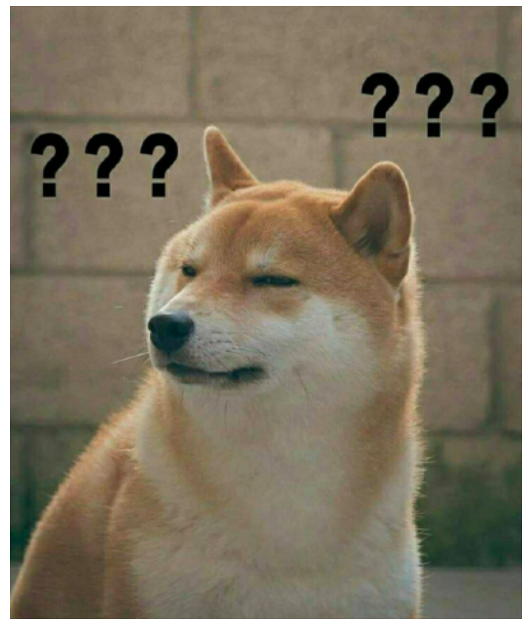


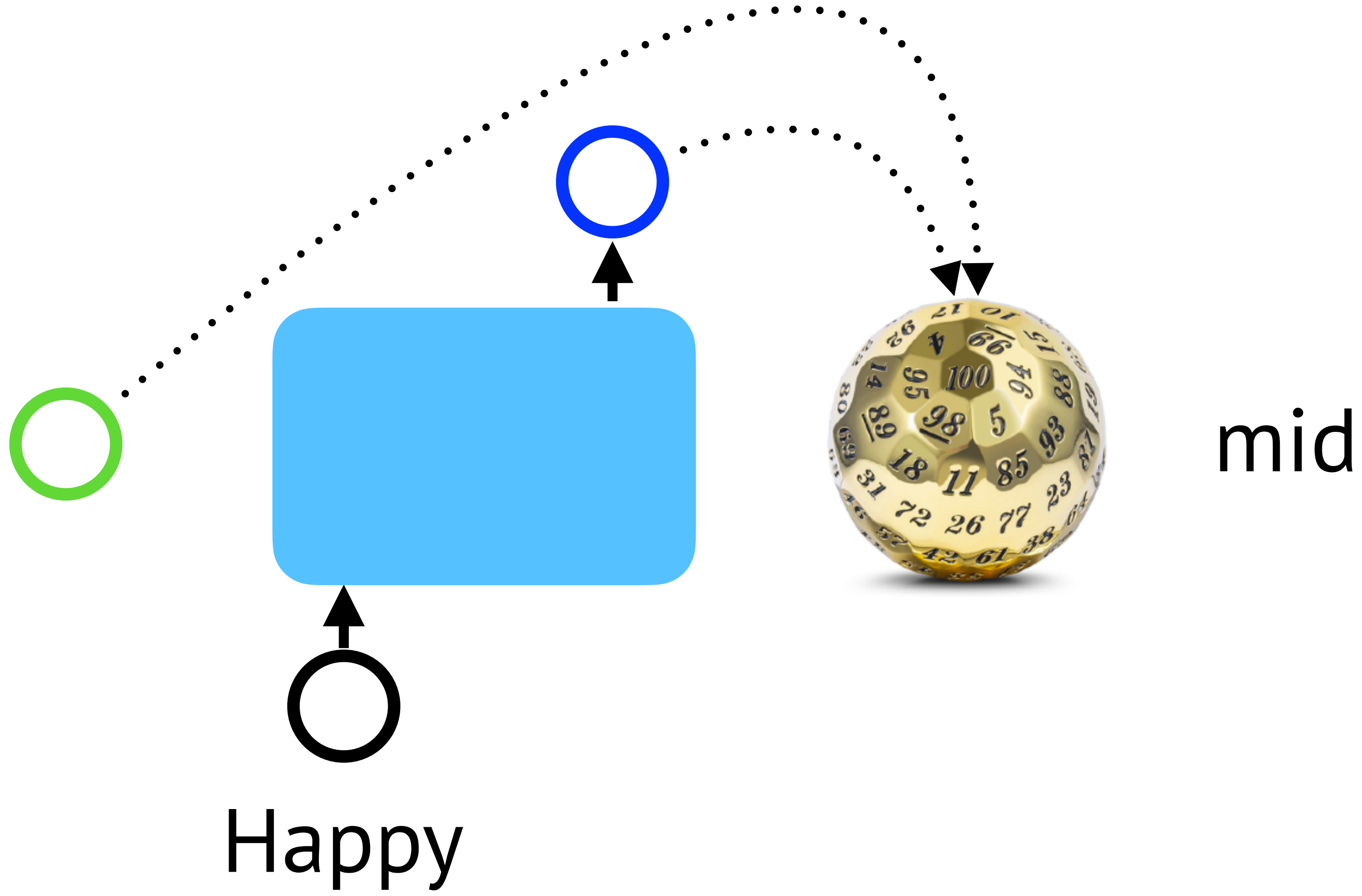
$p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x})$

Sequence to Sequence Model



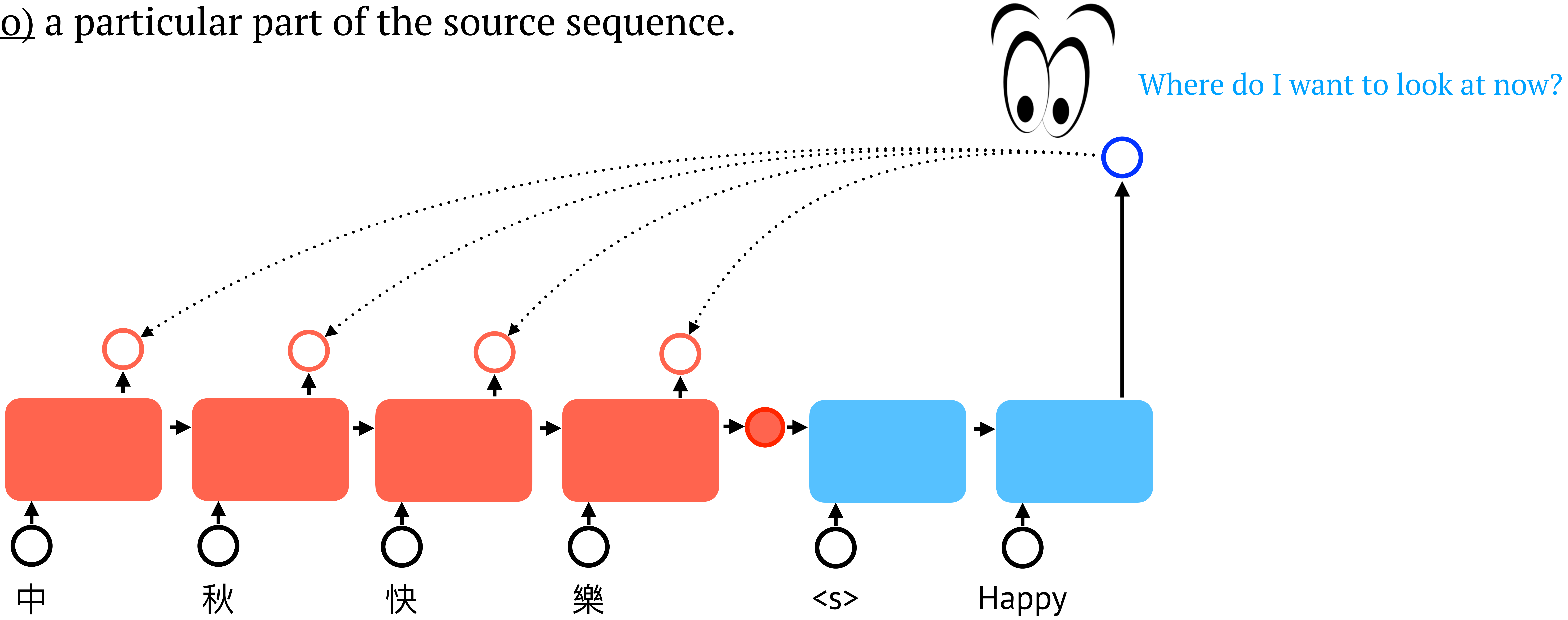
$p(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x})$





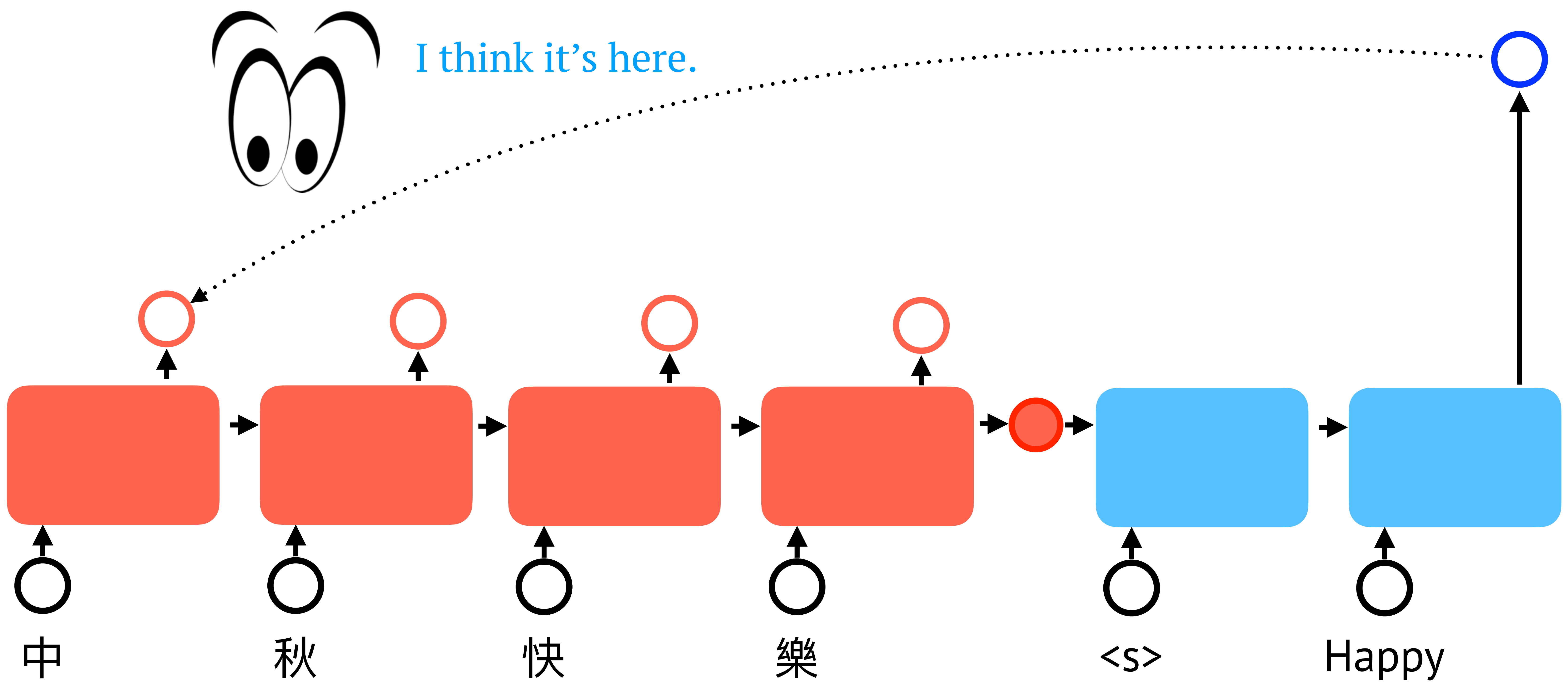
Attention Mechanism

Use direct connection to the encoder to focus on (attend to) a particular part of the source sequence.



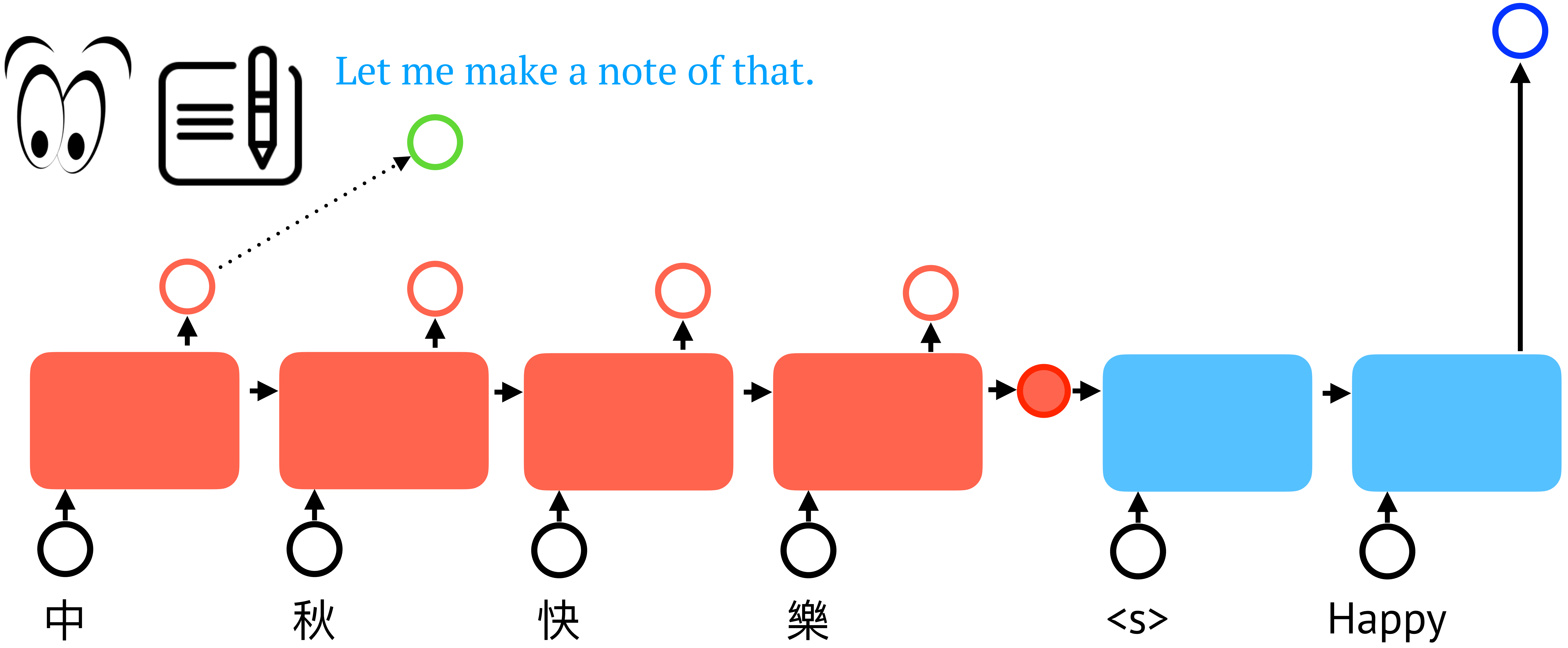
Attention Mechanism

Use direct connection to the encoder to focus on (attend to) a particular part of the source sequence.



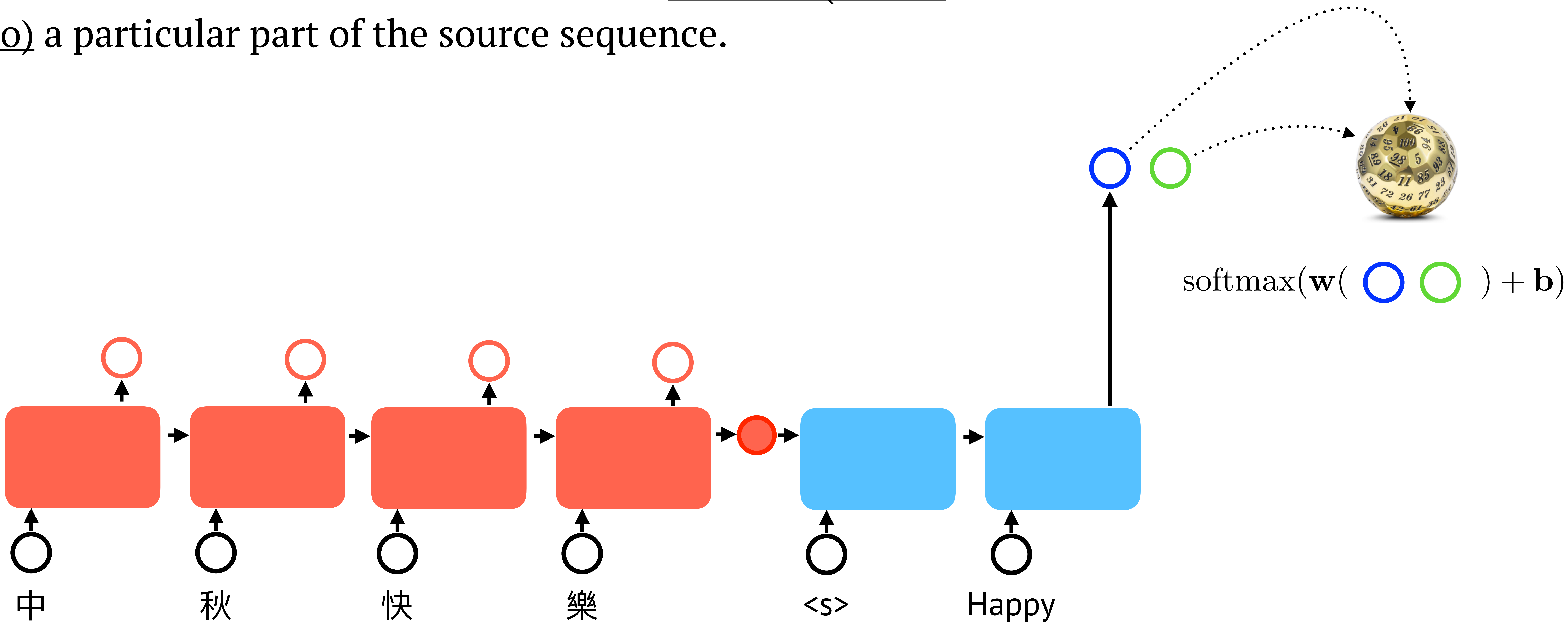
Attention Mechanism

Use direct connection to the encoder to focus on (attend to) a particular part of the source sequence.

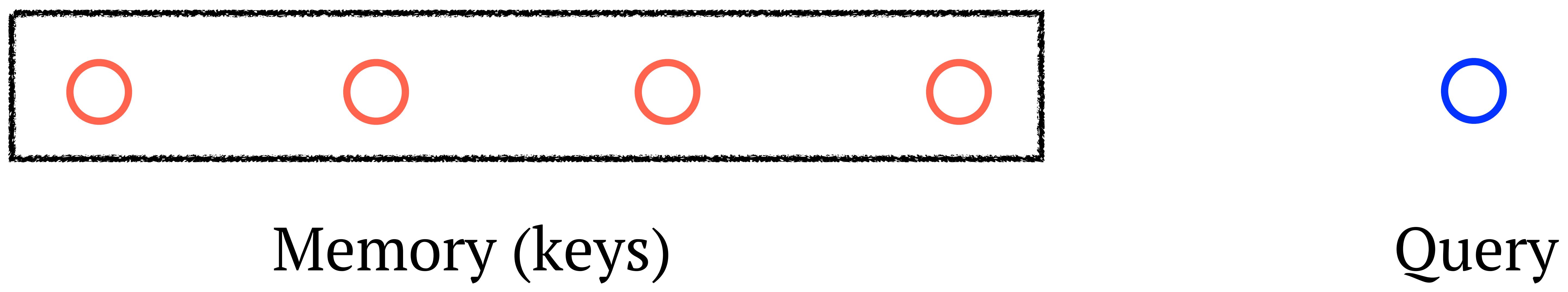


Attention Mechanism

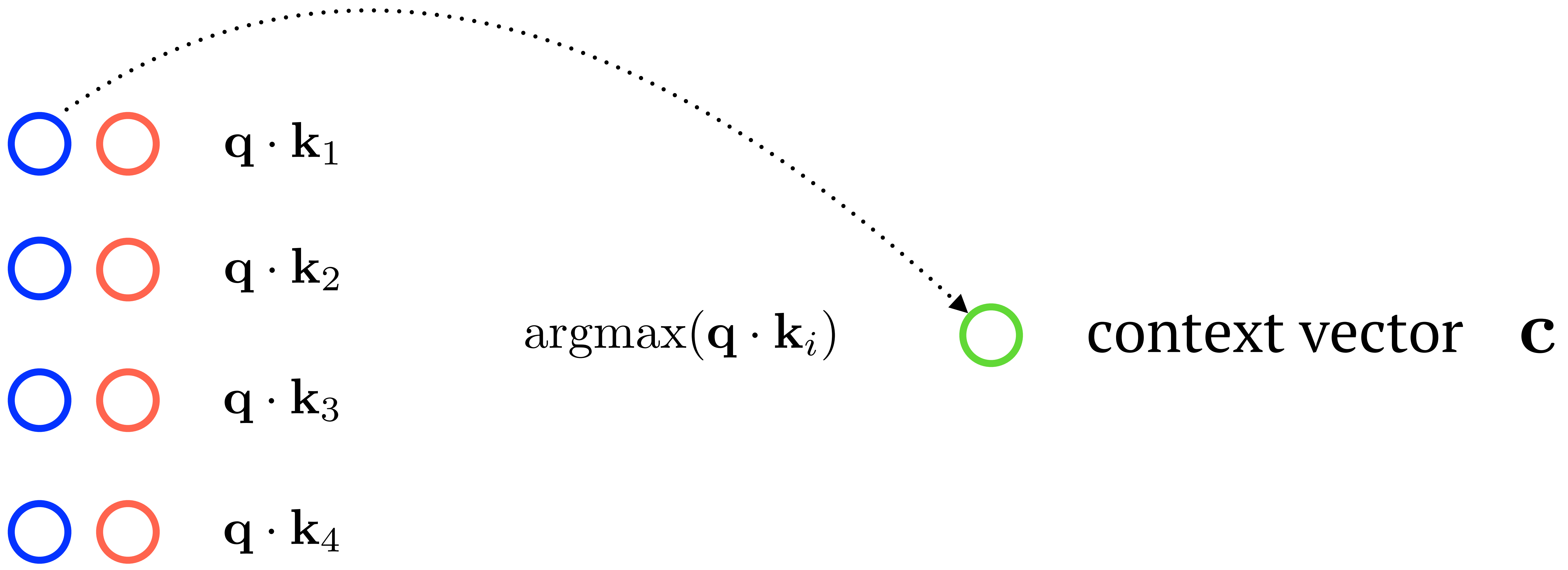
Use direct connection to the encoder to focus on (attend to) a particular part of the source sequence.



Memory Abstraction

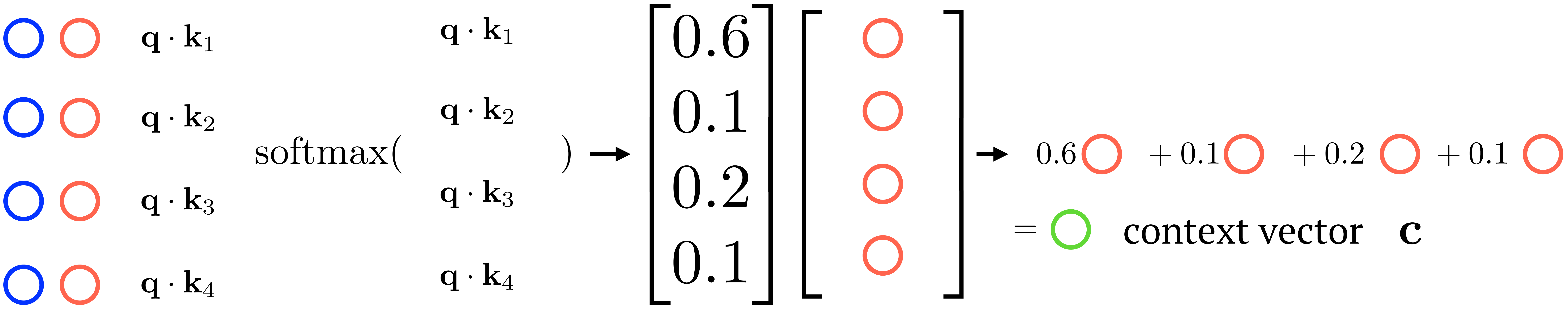


Task: Finding the most "relevant" item in the memory.



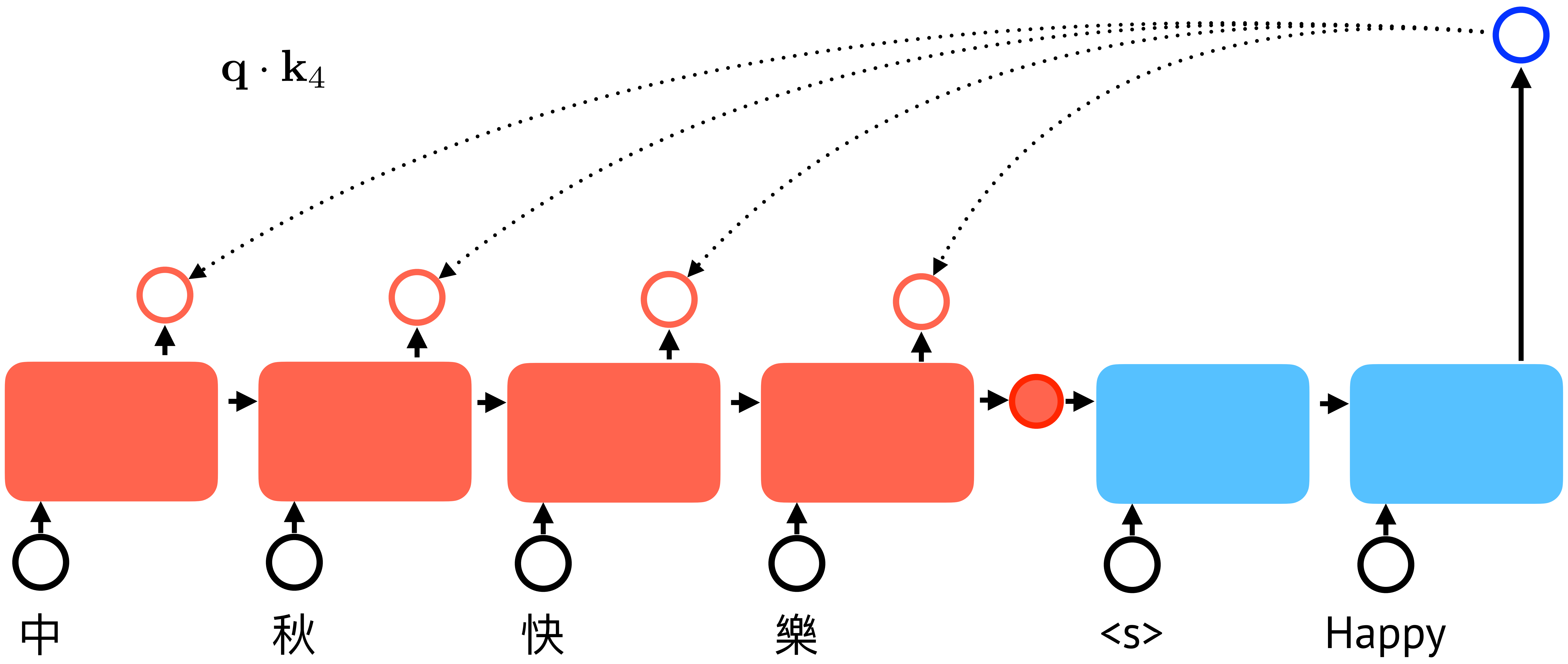
Dot-Product-Softmax Attention

Task: Finding the most "relevant" item in the memory.



Attention Mechanism

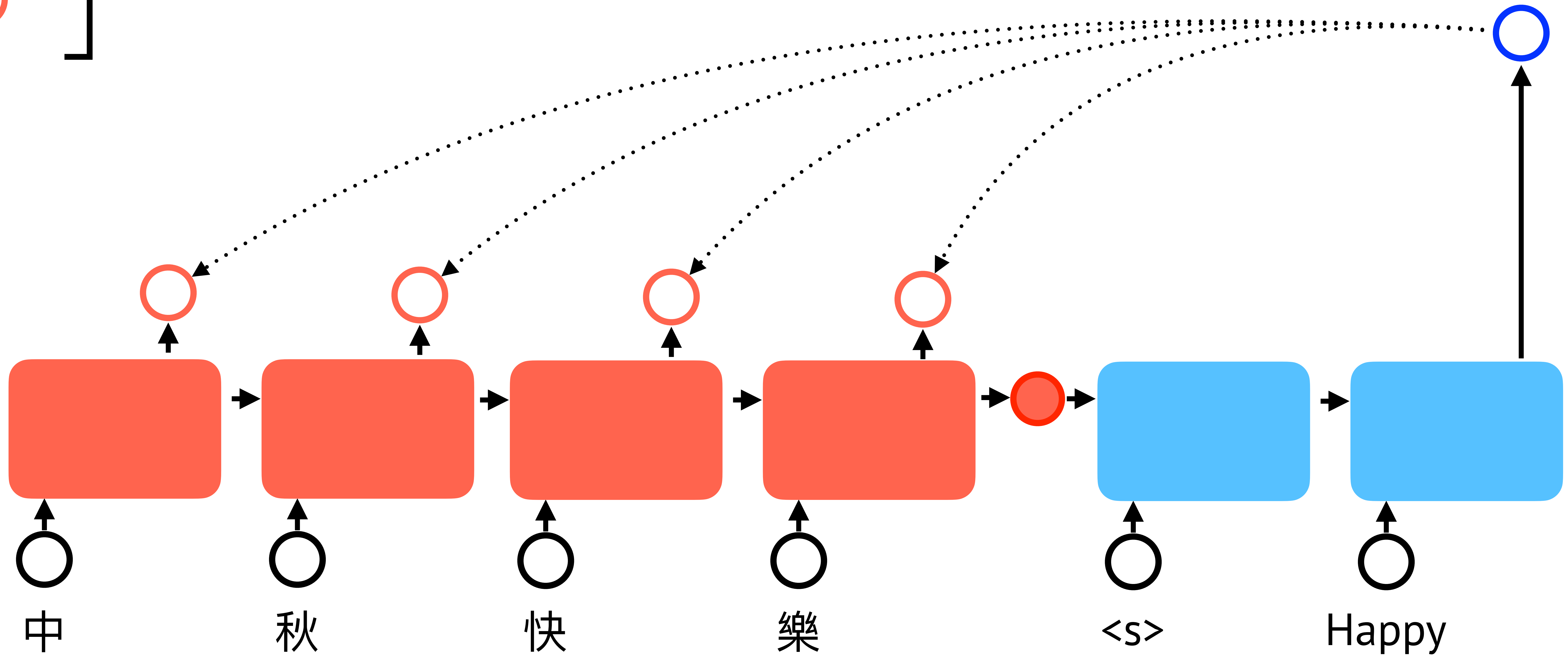
$$\text{softmax}\left(\begin{array}{c}\mathbf{q}\cdot\mathbf{k}_1 \\ \mathbf{q}\cdot\mathbf{k}_2 \\ \mathbf{q}\cdot\mathbf{k}_3 \\ \mathbf{q}\cdot\mathbf{k}_4\end{array}\right)$$



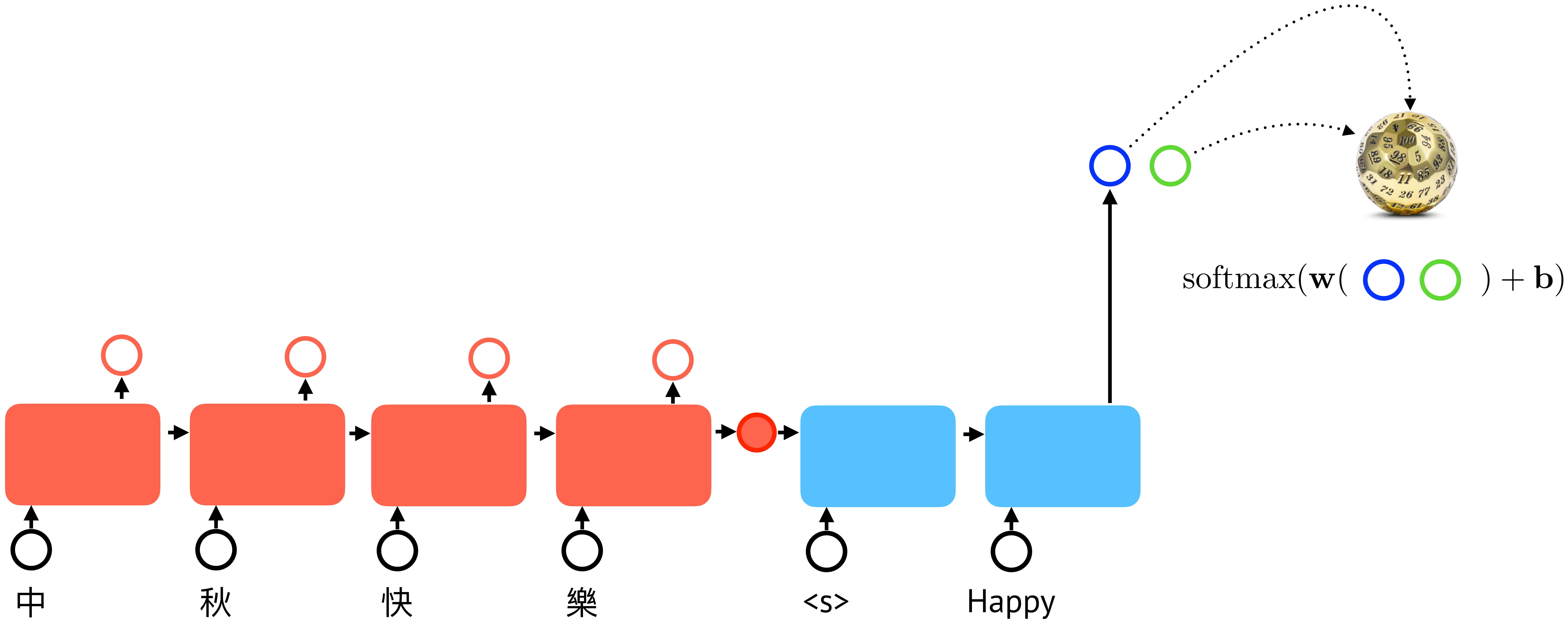
Attention Mechanism

$$\begin{bmatrix}0.6\\0.1\\0.2\\0.1\end{bmatrix}\begin{bmatrix}\bigcirc\\\bigcirc\\\bigcirc\\\bigcirc\end{bmatrix} \rightarrow 0.6\bigcirc + 0.1\bigcirc + 0.2\bigcirc + 0.1\bigcirc$$

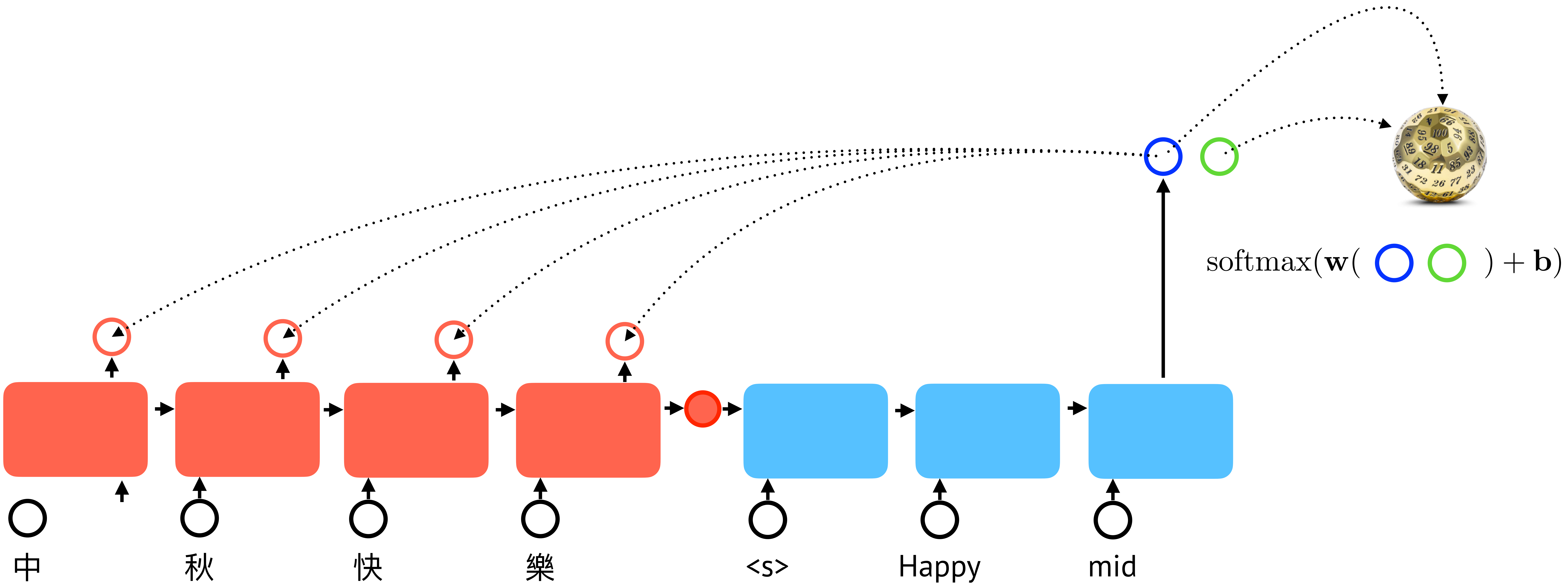
$$= \bigcirc \quad \text{context vector} \quad \mathbf{c}$$



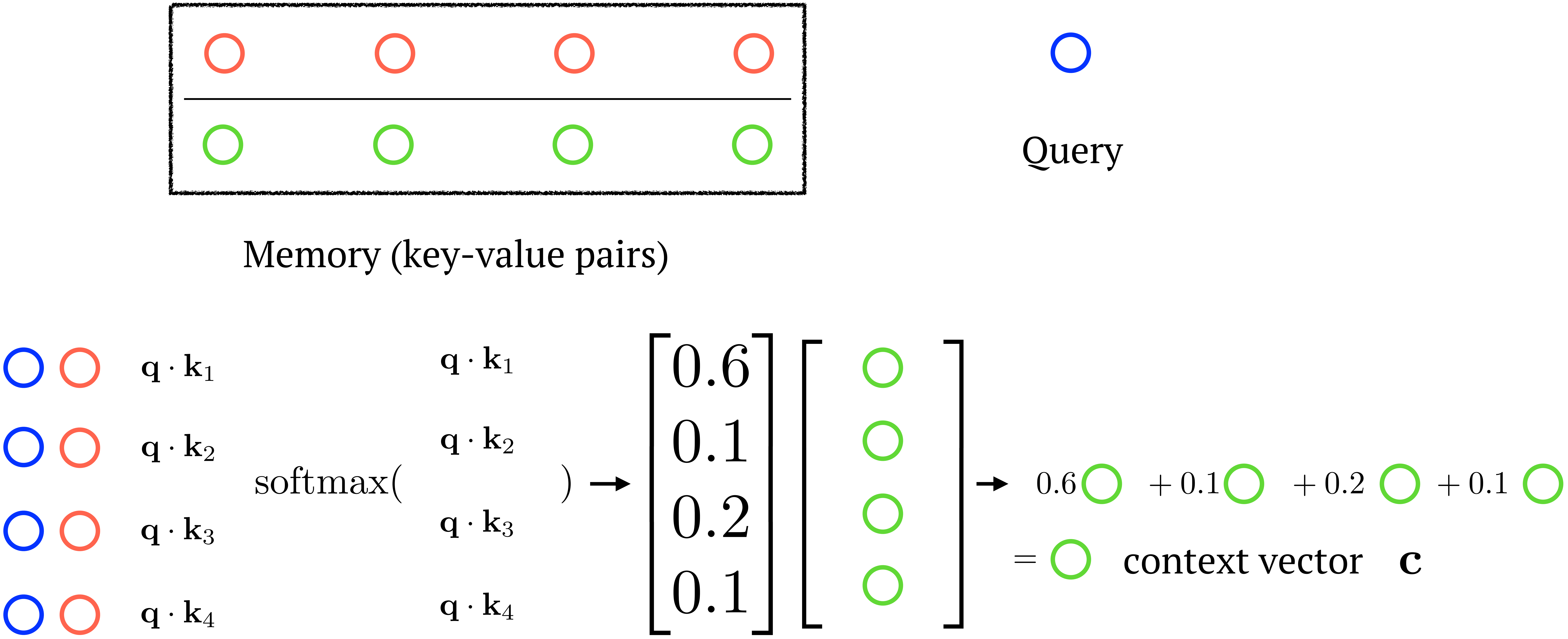
Attention Mechanism



Attention Mechanism



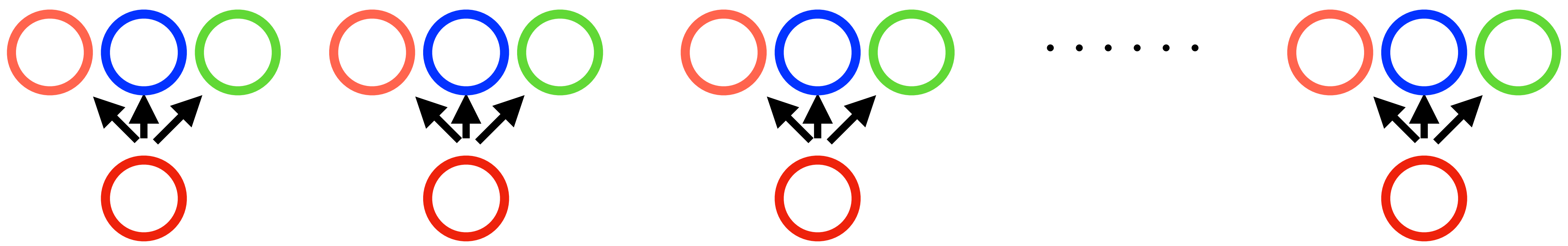
Dot-Product-Softmax Attention



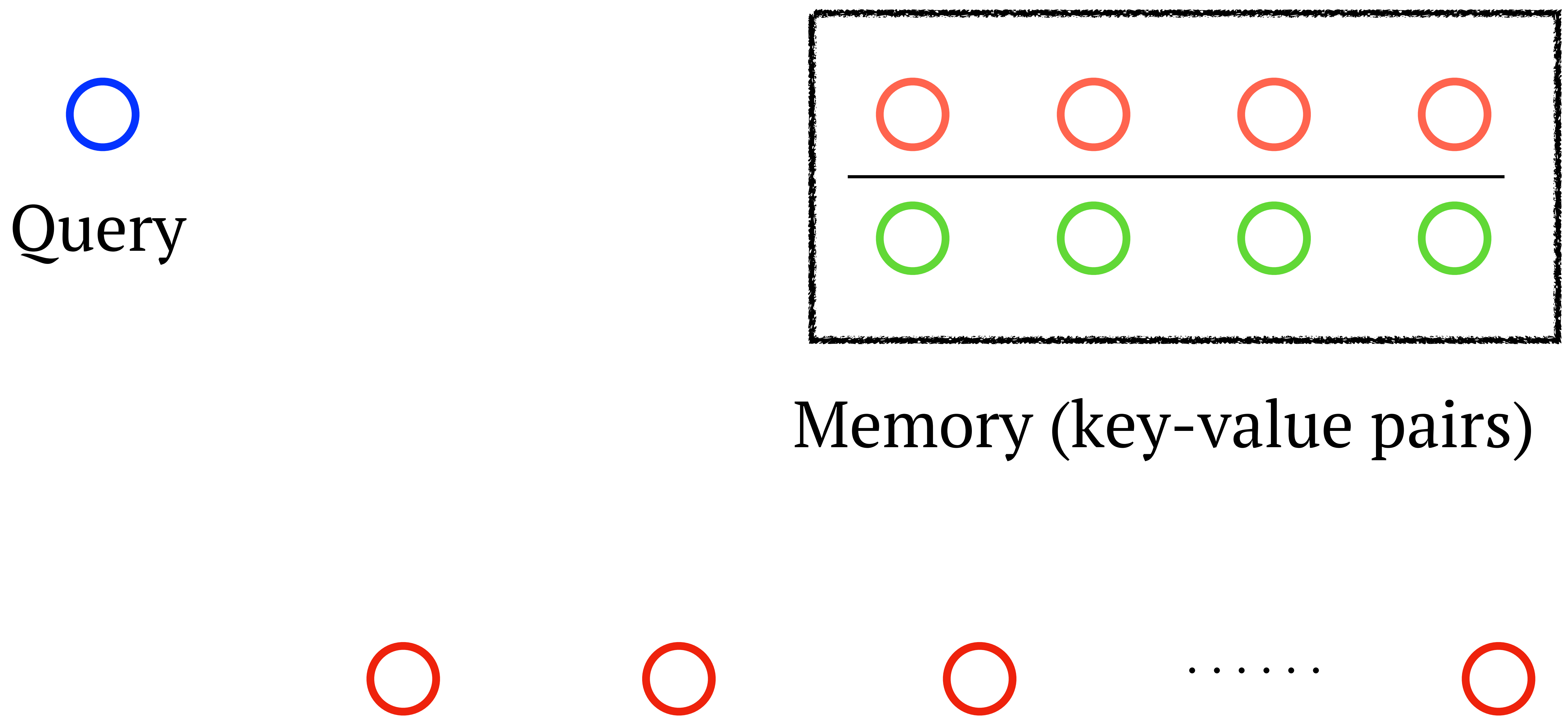
Dot-Product-Softmax Attention

$$\underbrace{\sum_{m=1}^{M} \underbrace{\frac{\overbrace{\exp\left(\boldsymbol{q}_n \boldsymbol{k}_m\right)}^{\text{similarity}}}{\sum_{m'=1}^{M} \exp\left(\boldsymbol{q}_n \boldsymbol{k}_{m'}\right)}}_{\text{normalized similarity}} \boldsymbol{v}_m^\top}_{\text{weighted sum}} = \mathbf{V}^\top \mathrm{softmax}(\mathbf{K}\boldsymbol{q}_n)$$

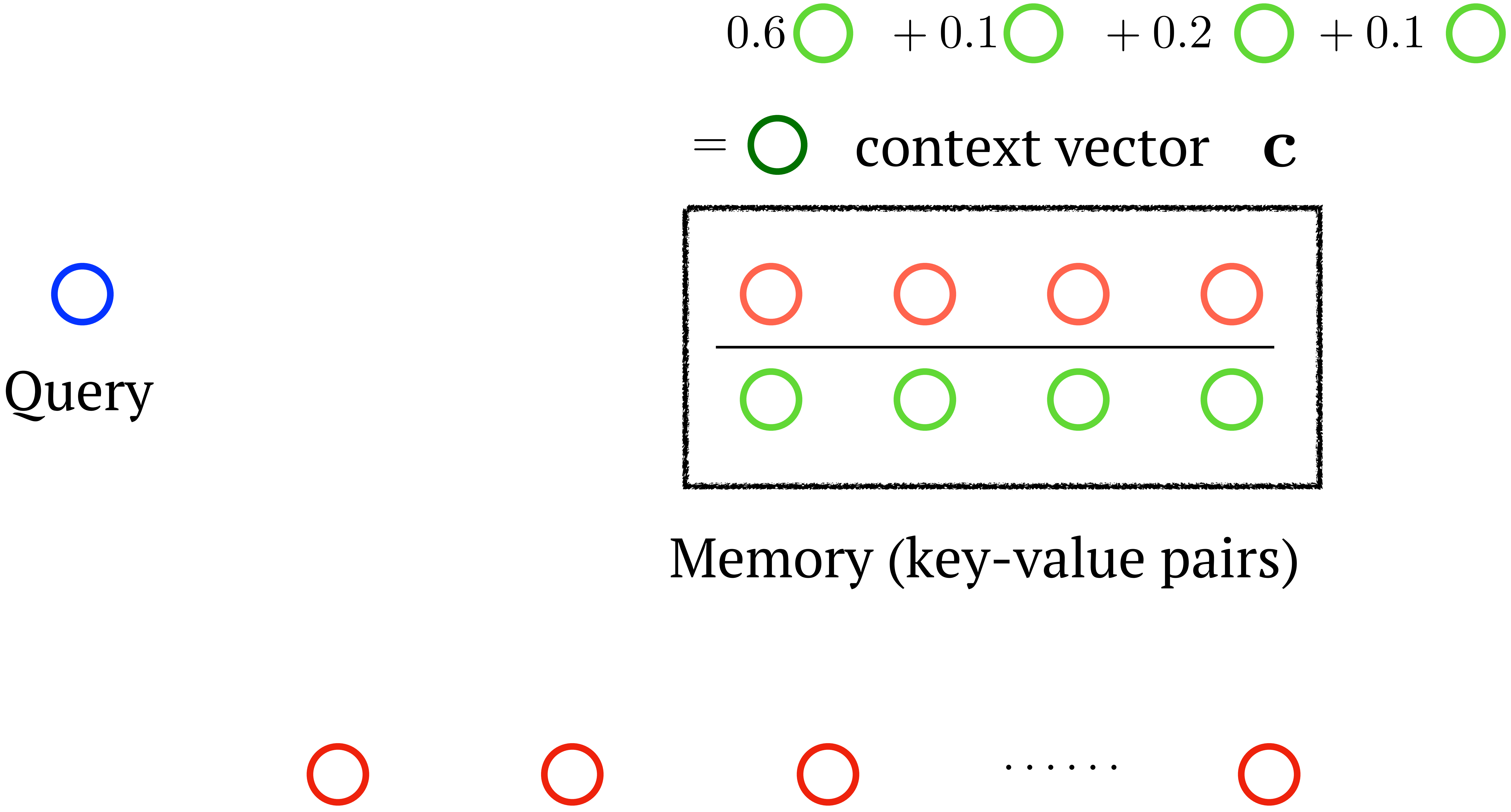
Considering the full sequence as context



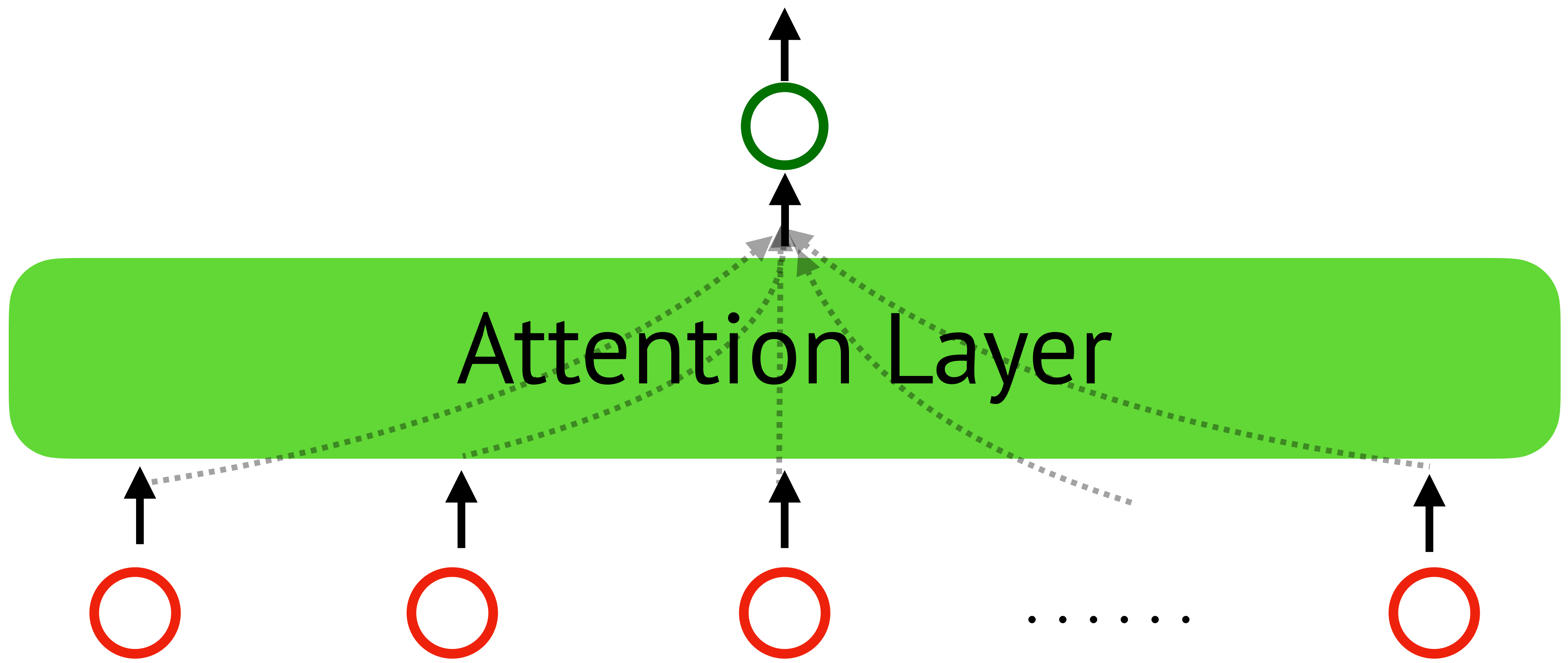
Attention Mechanism



Attention Mechanism

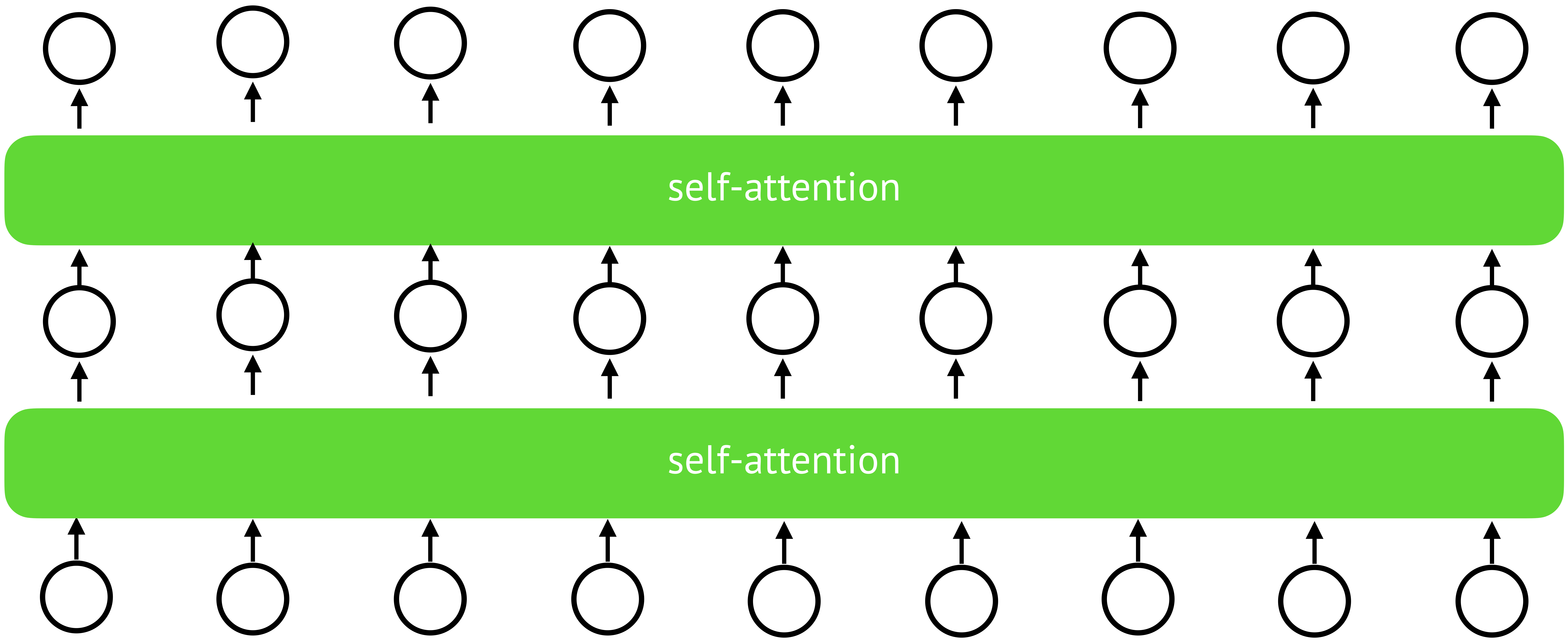


Self-attention

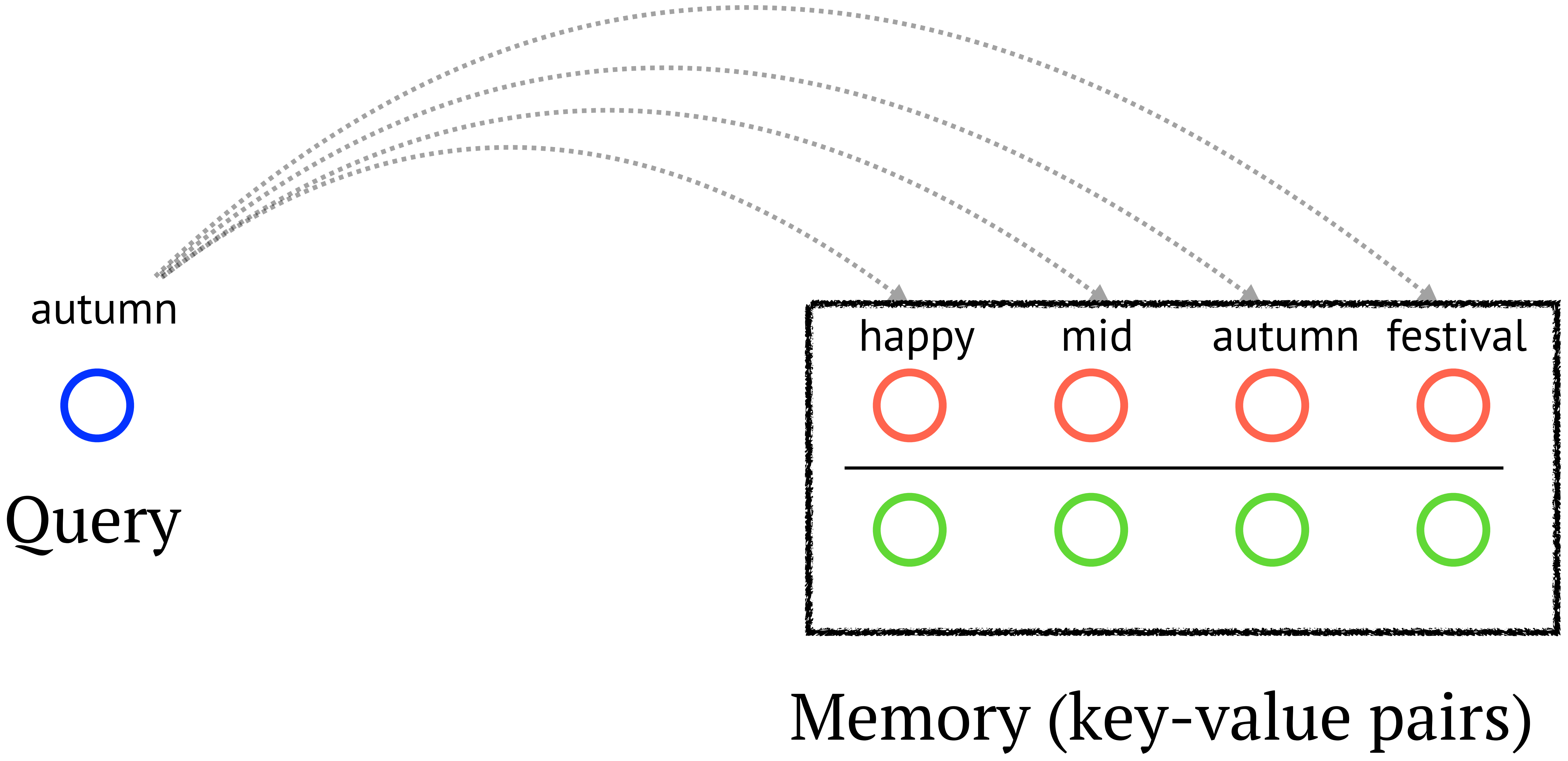


This is almost transformer — except a few things.

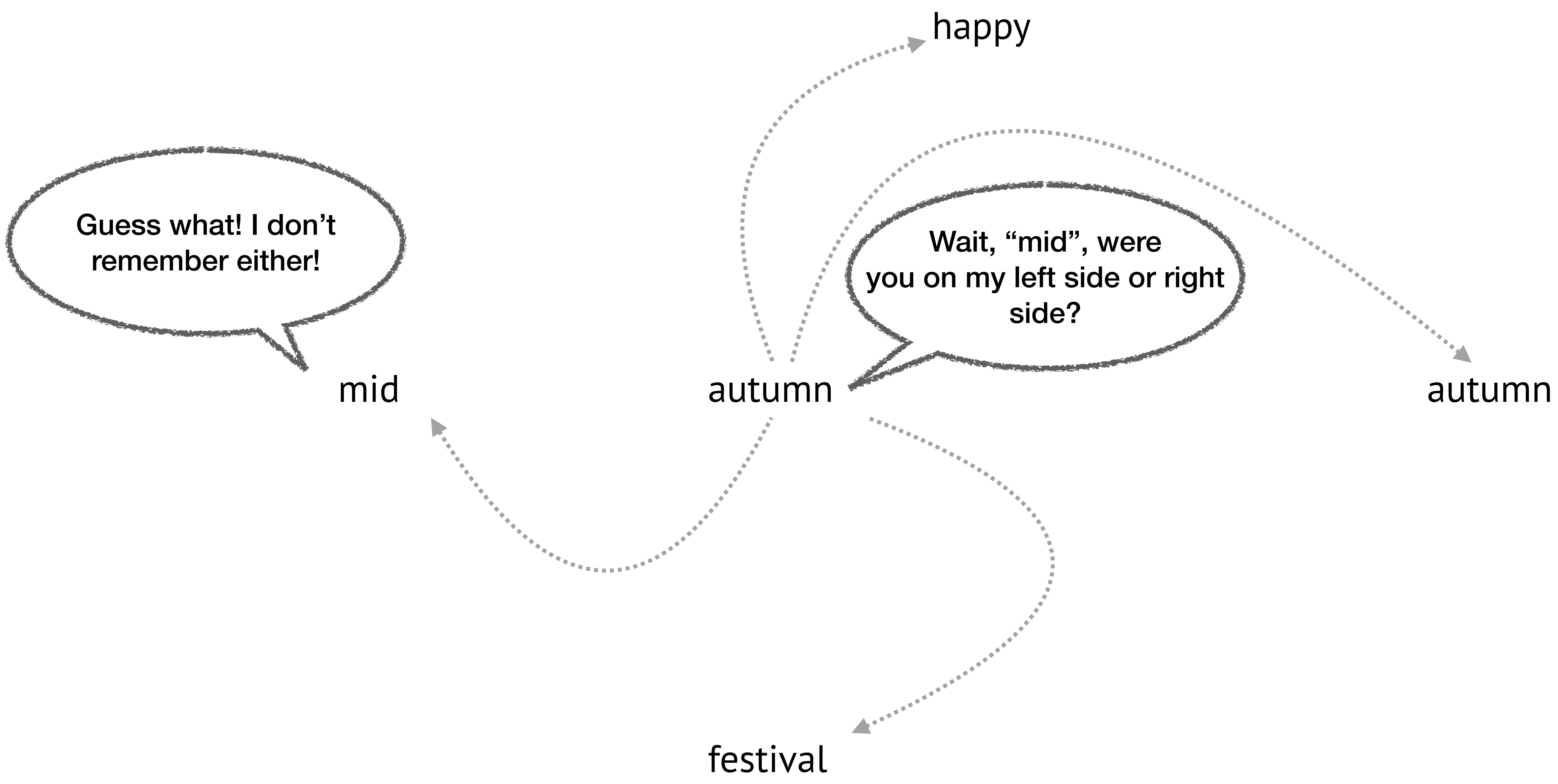
Transformer (almost)



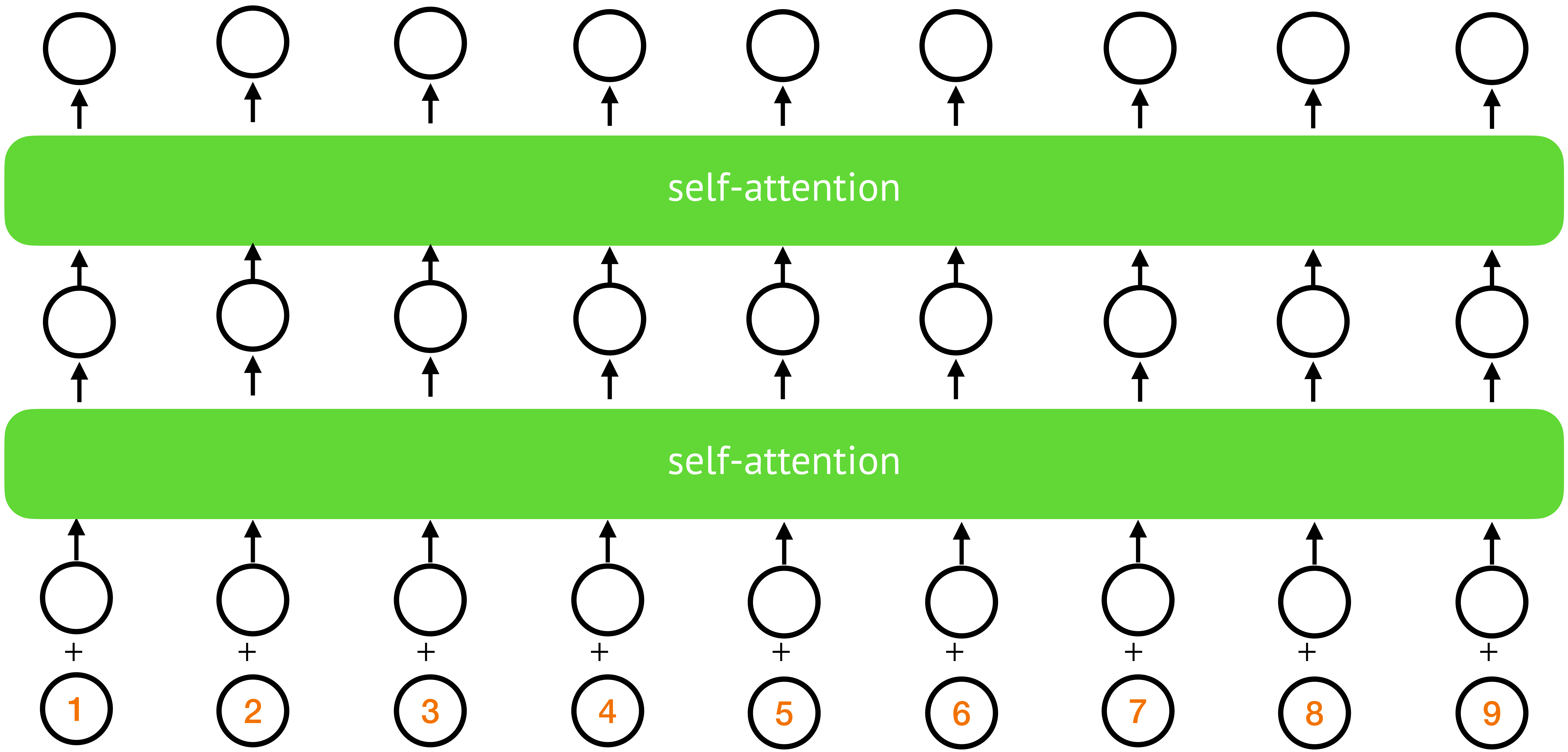
Self-attention in Transformer



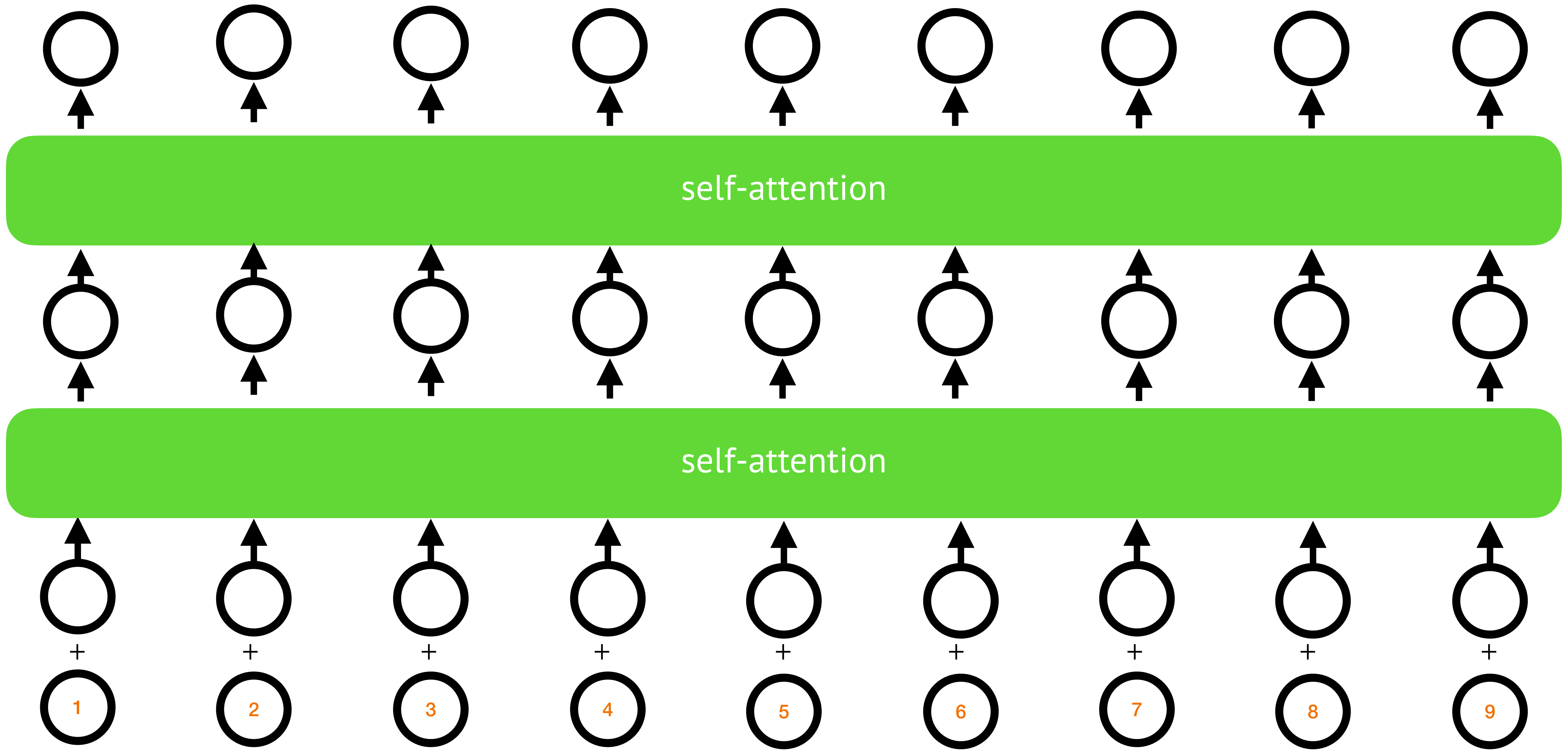
Self-attention in Transformer



Positional Embeddings

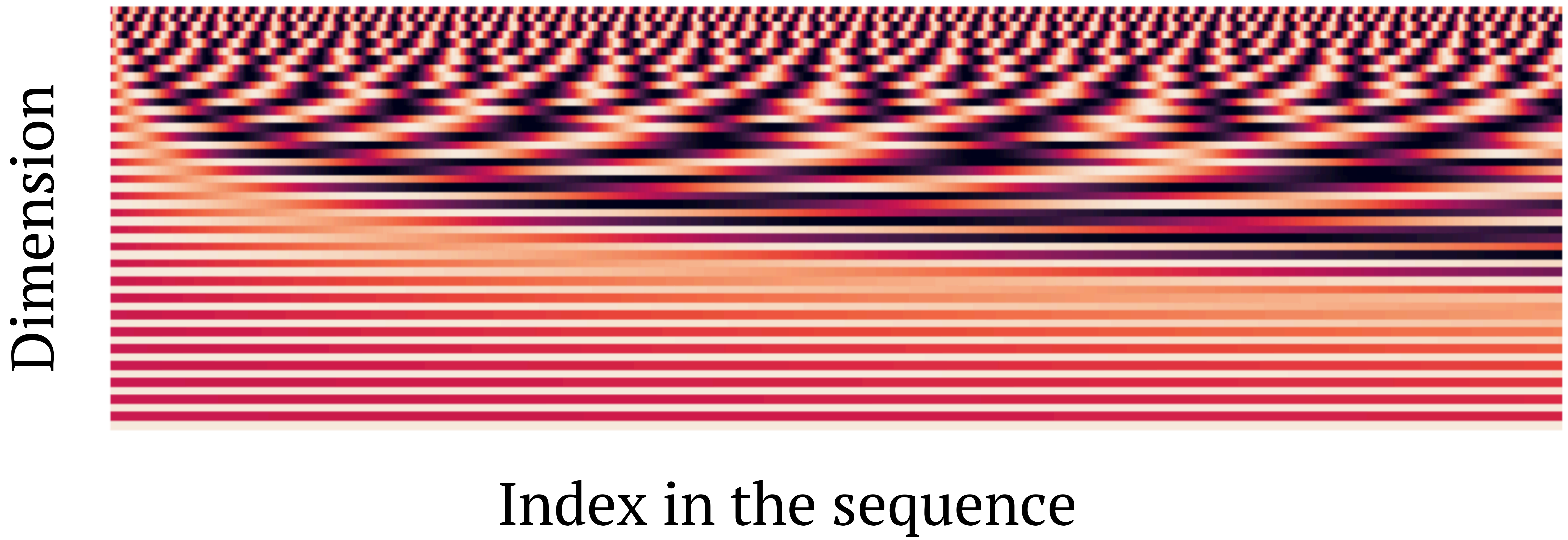


Transformer (positional embedding)



Positional Encoding

$$
\begin{bmatrix}
\sin\left(\frac{i}{10000^{2\times\frac{1}{d}}}\right) \\
\cos\left(\frac{i}{10000^{2\times\frac{1}{d}}}\right) \\
\vdots \\
\sin\left(\frac{i}{10000^{2\times\frac{d/2}{d}}}\right) \\
\cos\left(\frac{i}{10000^{2\times\frac{d/2}{d}}}\right)
\end{bmatrix}
$$

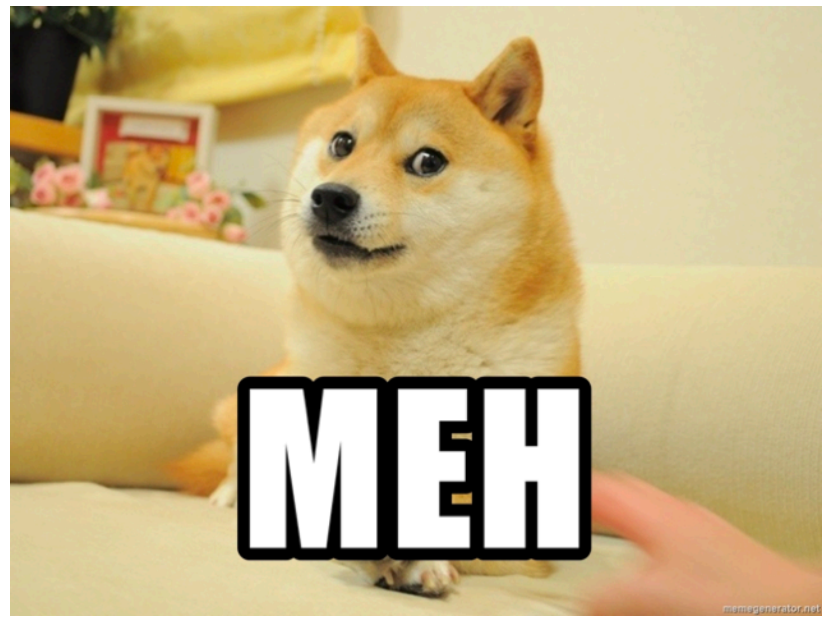


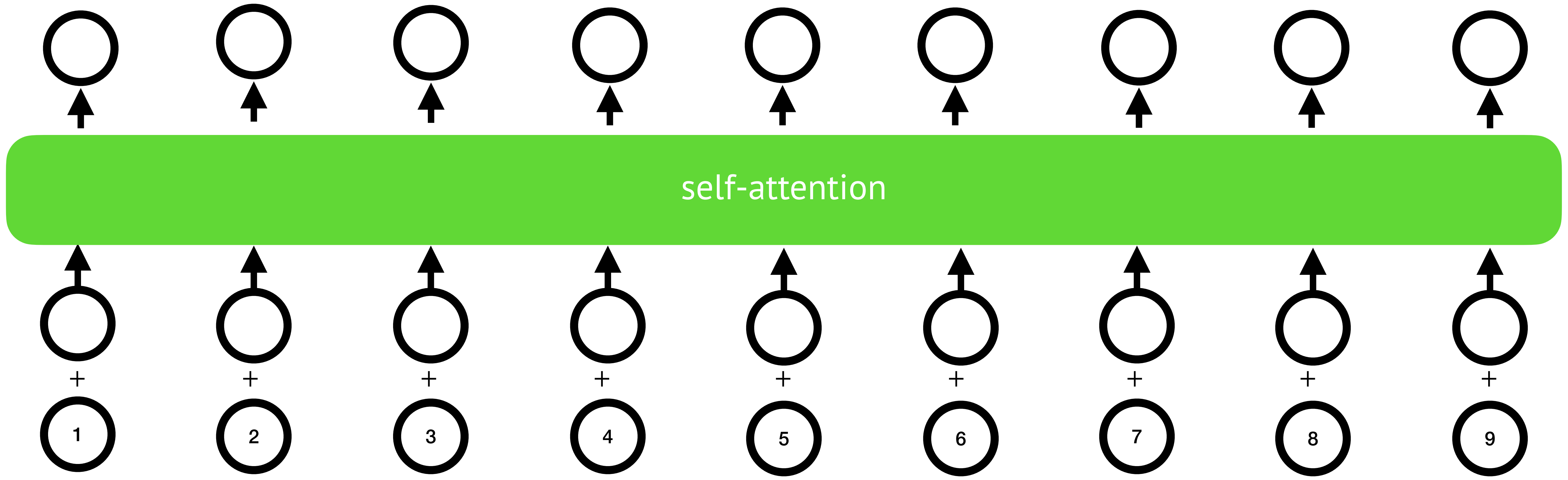
The idea of relative position

Positional Encoding

$$\begin{bmatrix} \sin\left(\frac{i}{10000^{2\times\frac{1}{d}}}\right) \\ \cos\left(\frac{i}{10000^{2\times\frac{1}{d}}}\right) \\ \vdots \\ \sin\left(\frac{i}{10000^{2\times\frac{d/2}{d}}}\right) \\ \cos\left(\frac{i}{10000^{2\times\frac{d/2}{d}}}\right) \end{bmatrix}$$

Periodic: Hope this will work in extrapolation.

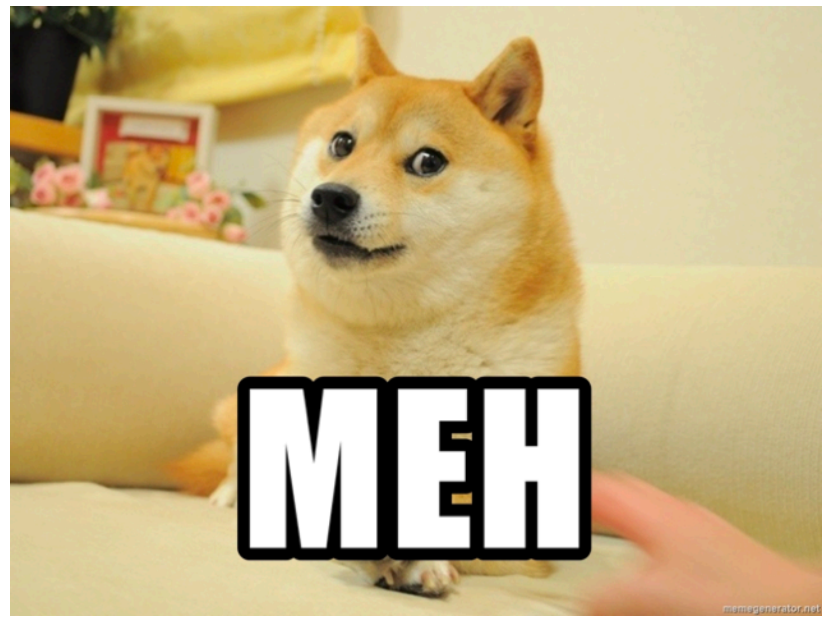


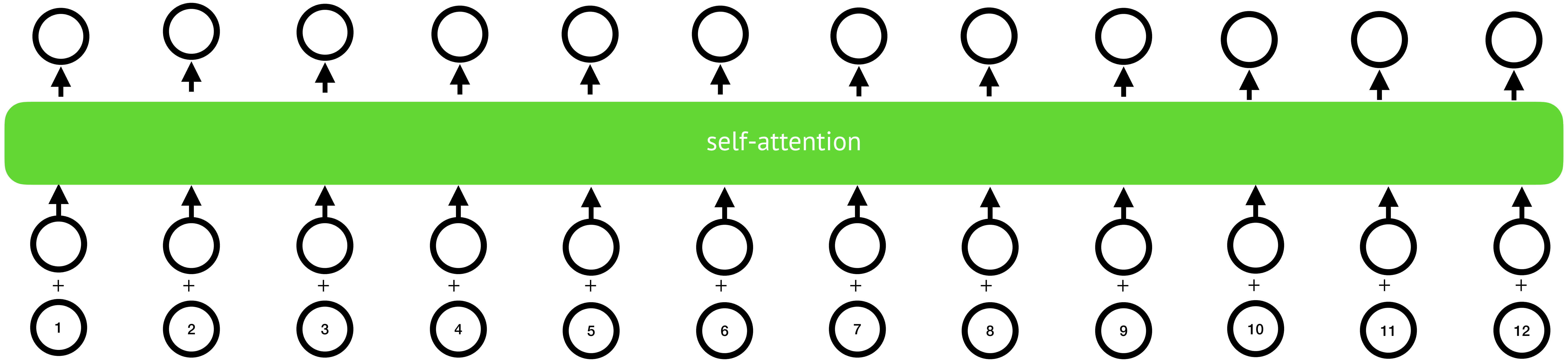


Positional Encoding

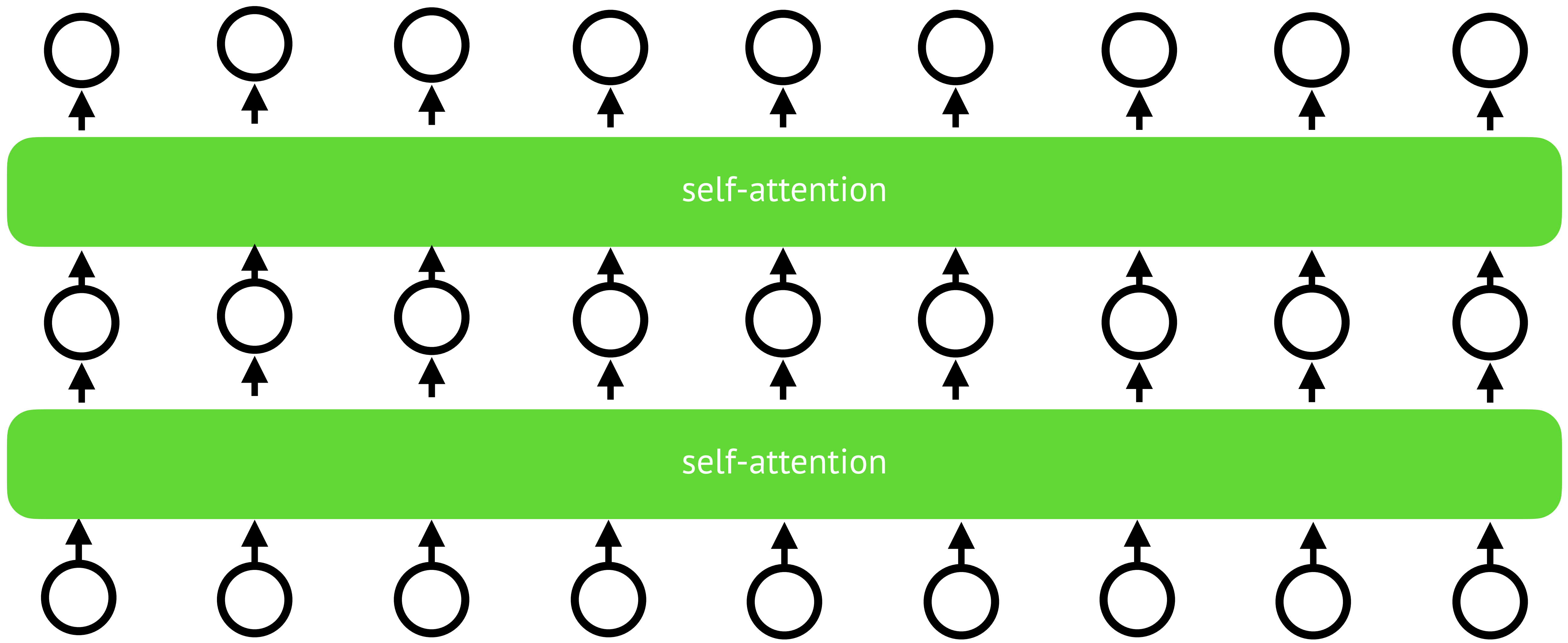
$$\begin{bmatrix} \sin\left(\frac{i}{10000^{2\times\frac{1}{d}}}\right) \\ \cos\left(\frac{i}{10000^{2\times\frac{1}{d}}}\right) \\ \vdots \\ \sin\left(\frac{i}{10000^{2\times\frac{d/2}{d}}}\right) \\ \cos\left(\frac{i}{10000^{2\times\frac{d/2}{d}}}\right) \end{bmatrix}$$

Periodic: Hope this will work in extrapolation. (No)

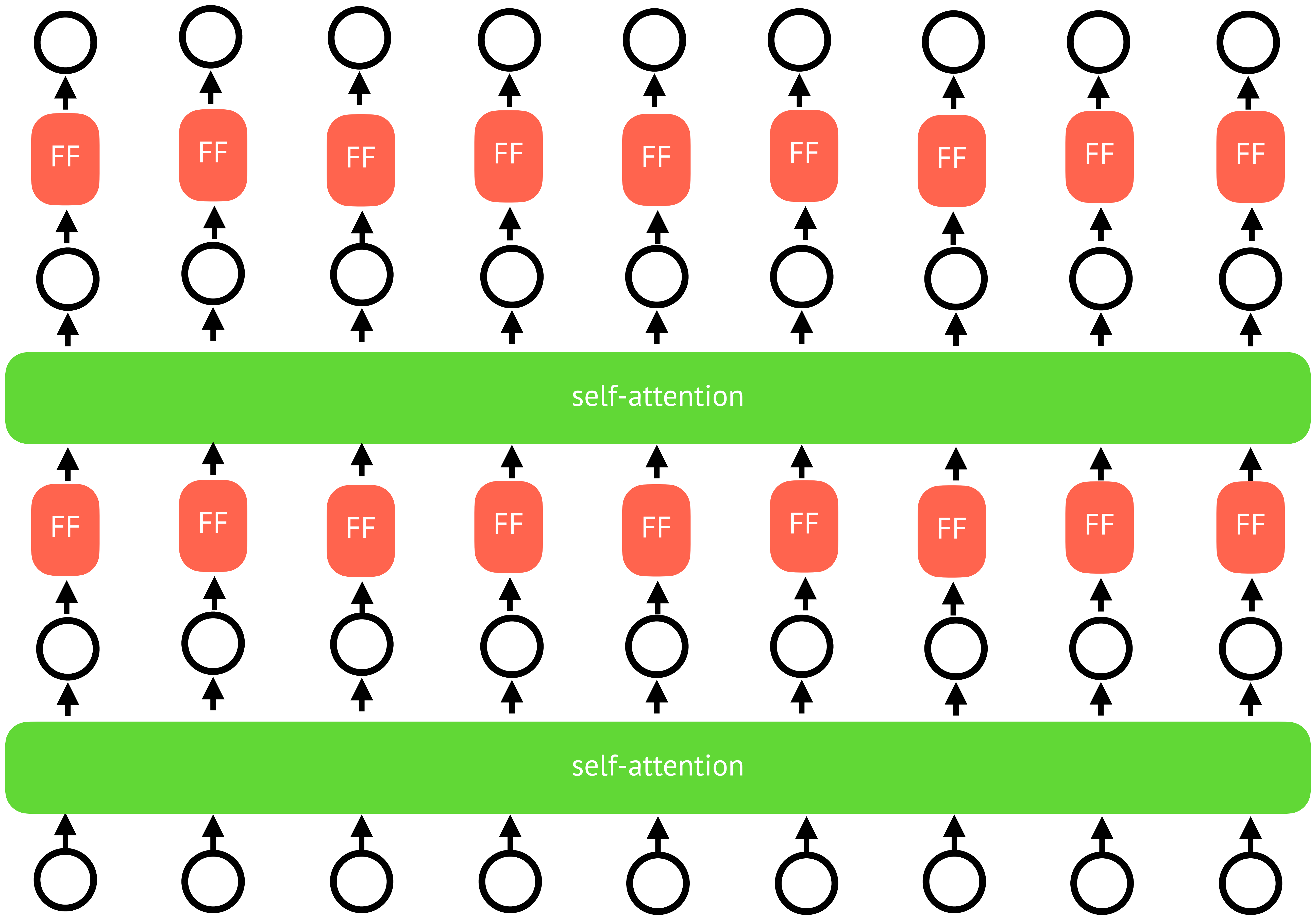




Feed Forward Layer



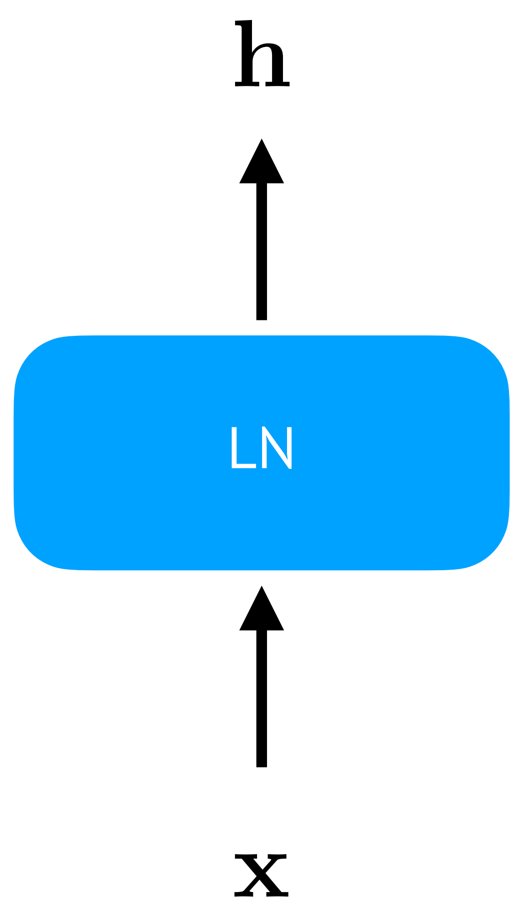
Feed Forward Layer



Layer Normalization (Ba et al, 2016)

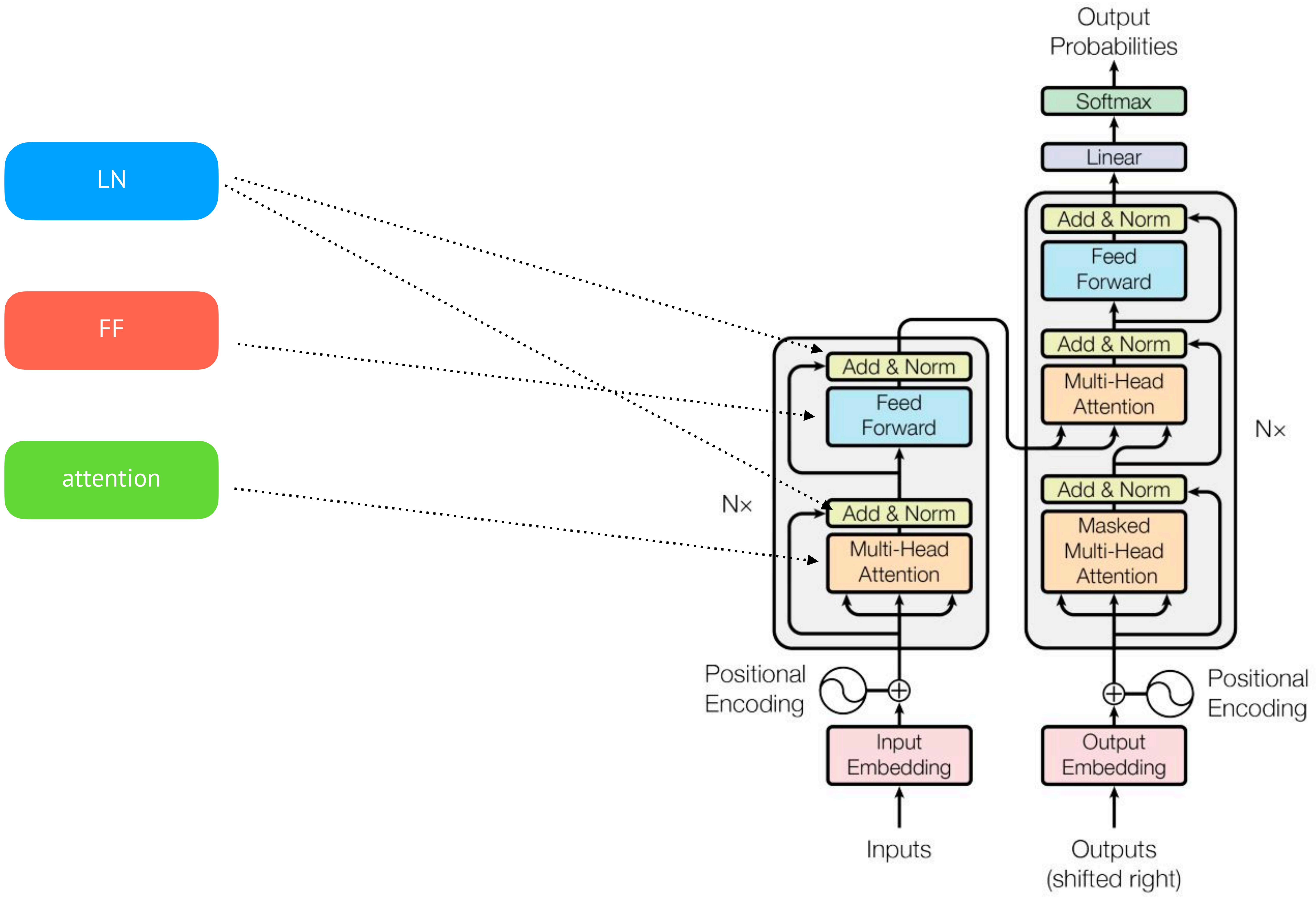
$$\mathbf{h} = \mathbf{g} \odot N(\mathbf{x}) + \mathbf{b}$$

$$N(\mathbf{x}) = \frac{\mathbf{x} - \mu}{\sigma} \qquad \mu = \frac{1}{H}\sum_{i=1}^{H} x_i \qquad \sigma = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(x_i - \mu)^2}$$

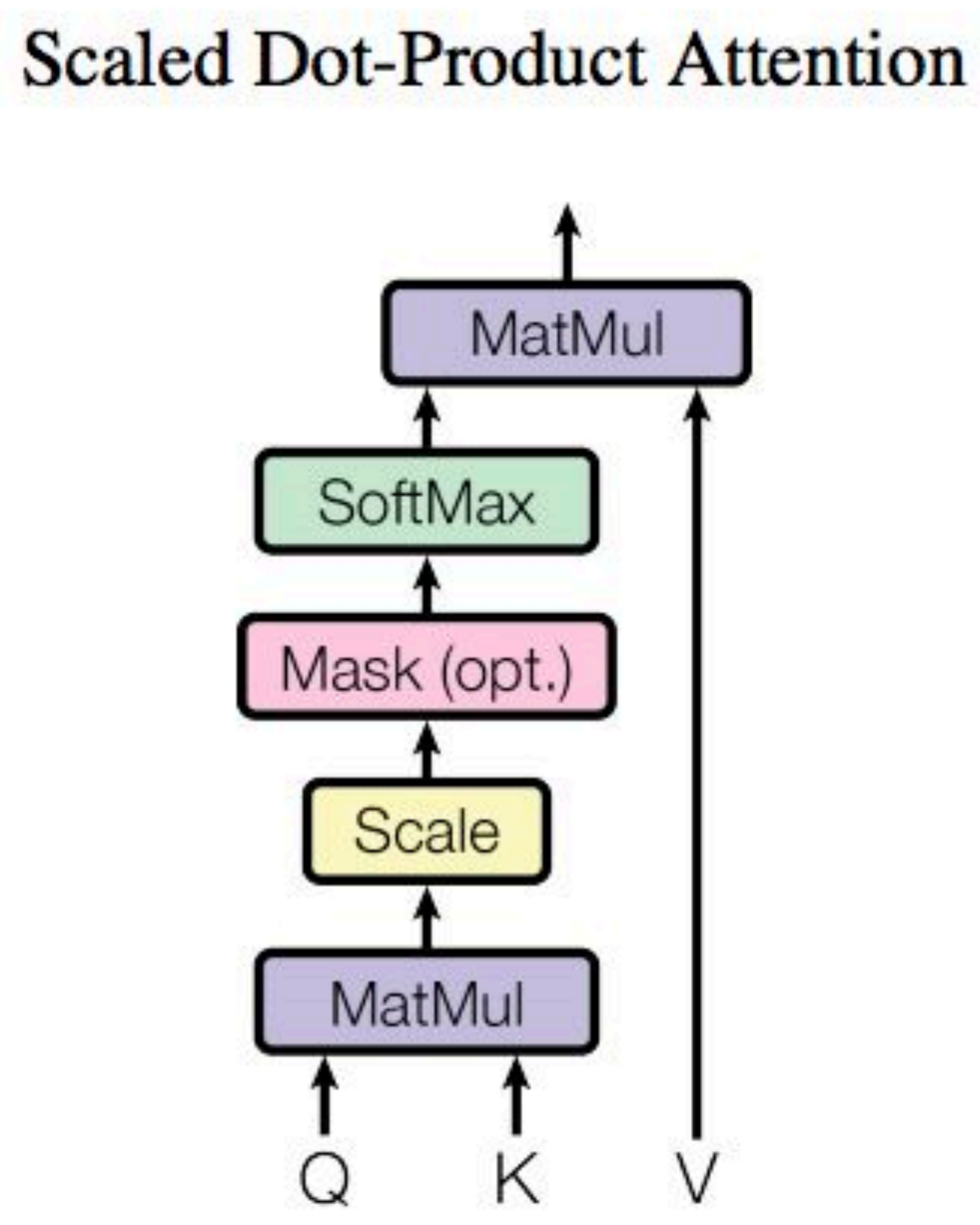


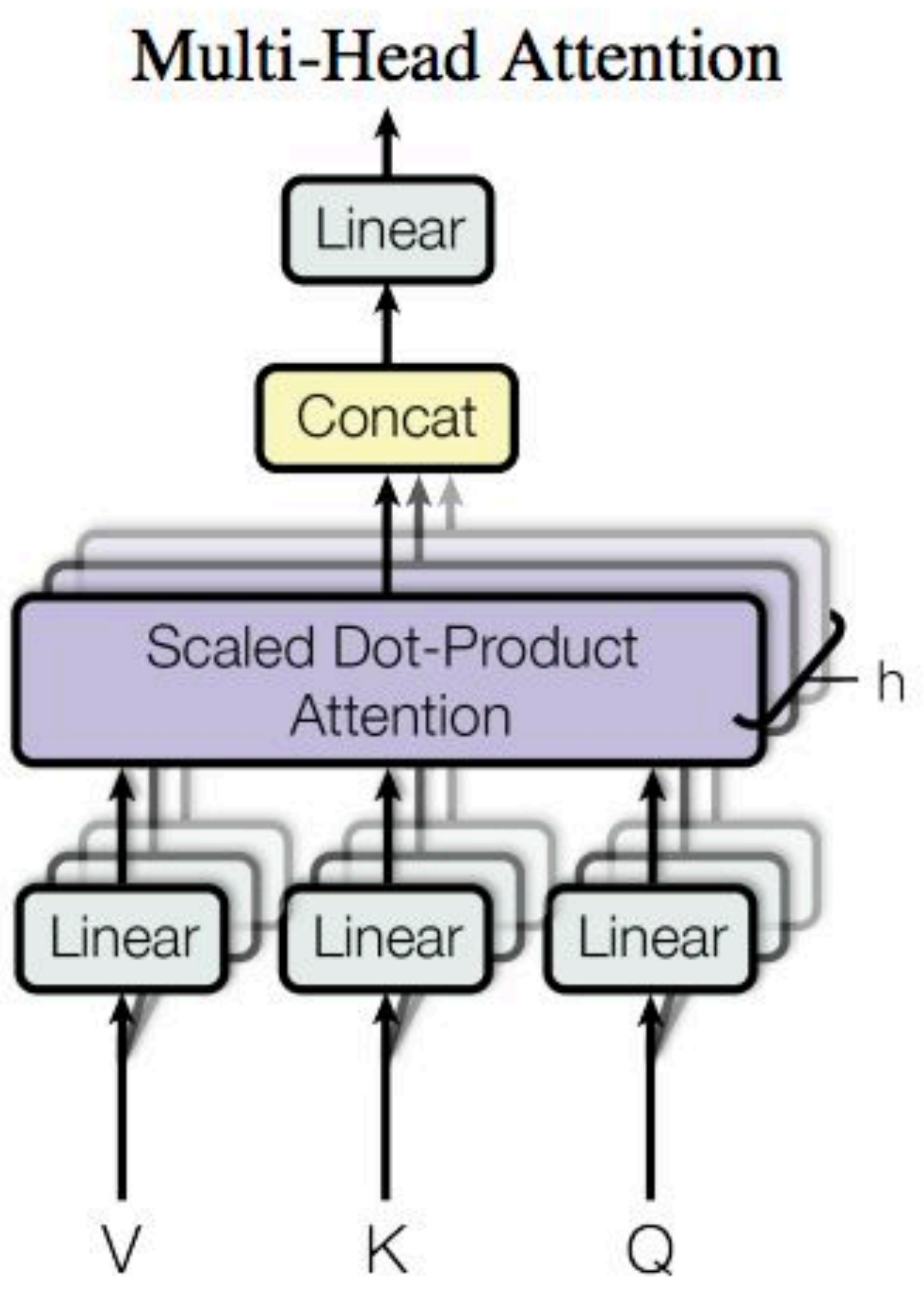
Smoother gradients, faster training and better generalization accuracy. (Xu et al, Neurips 2019)

Layer Normalization



Multi-head Attention

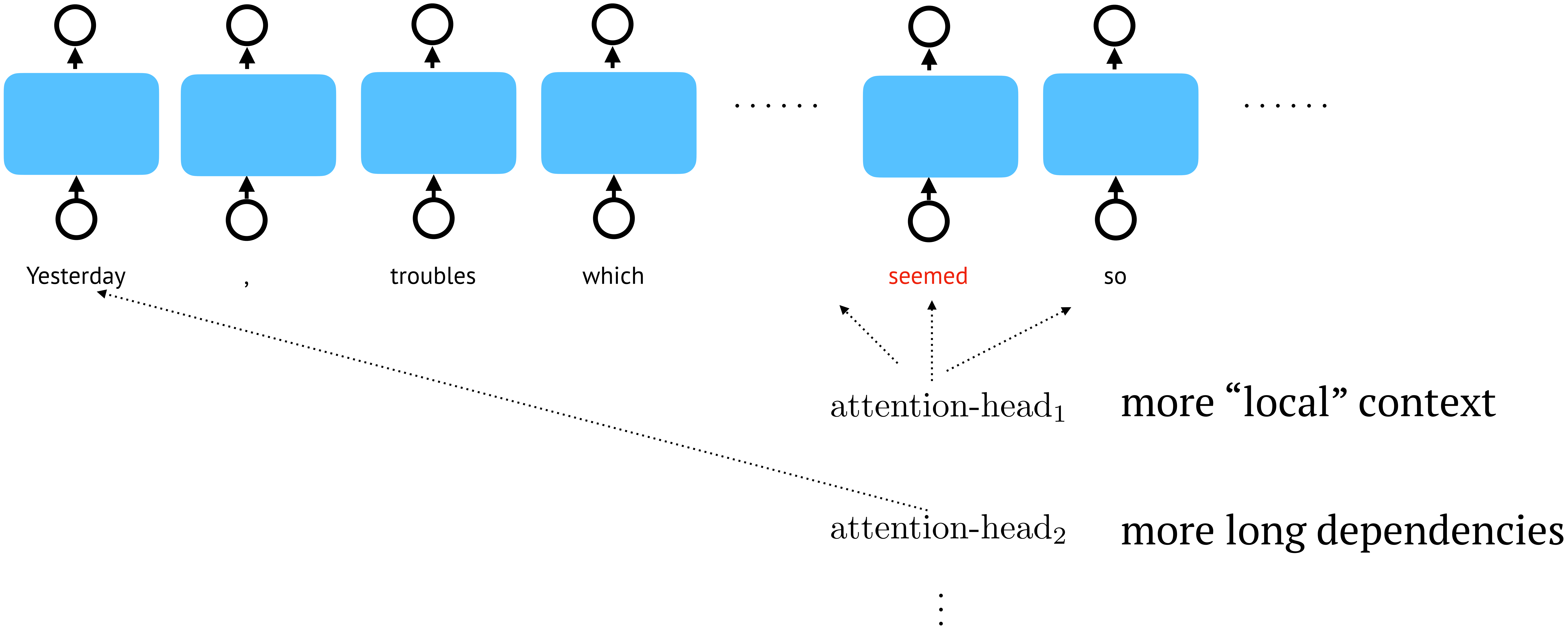




$$\text{score}(q, k) = \frac{q^T k}{\sqrt{d_k}}$$

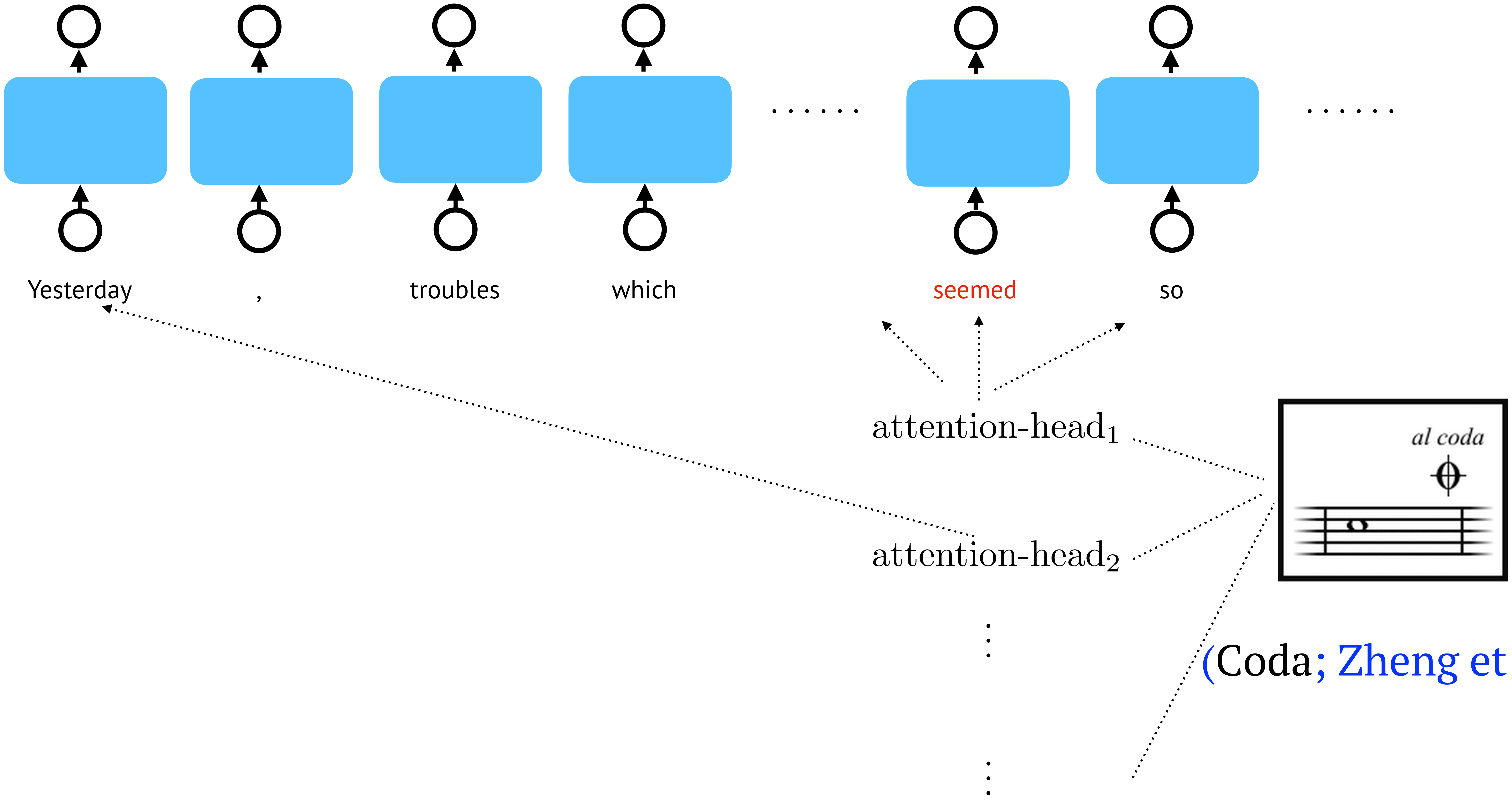
multiple copies

Multi-head Attention



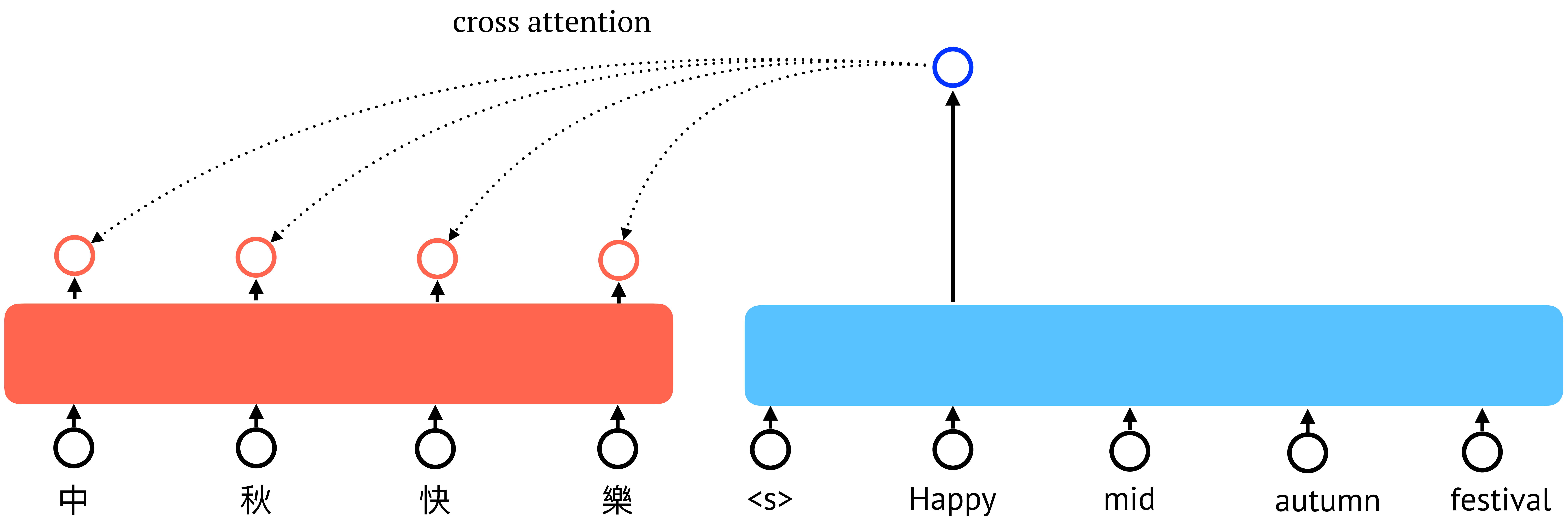
Improve the “resolution” of the attention mechanism.

Multi-head Attention

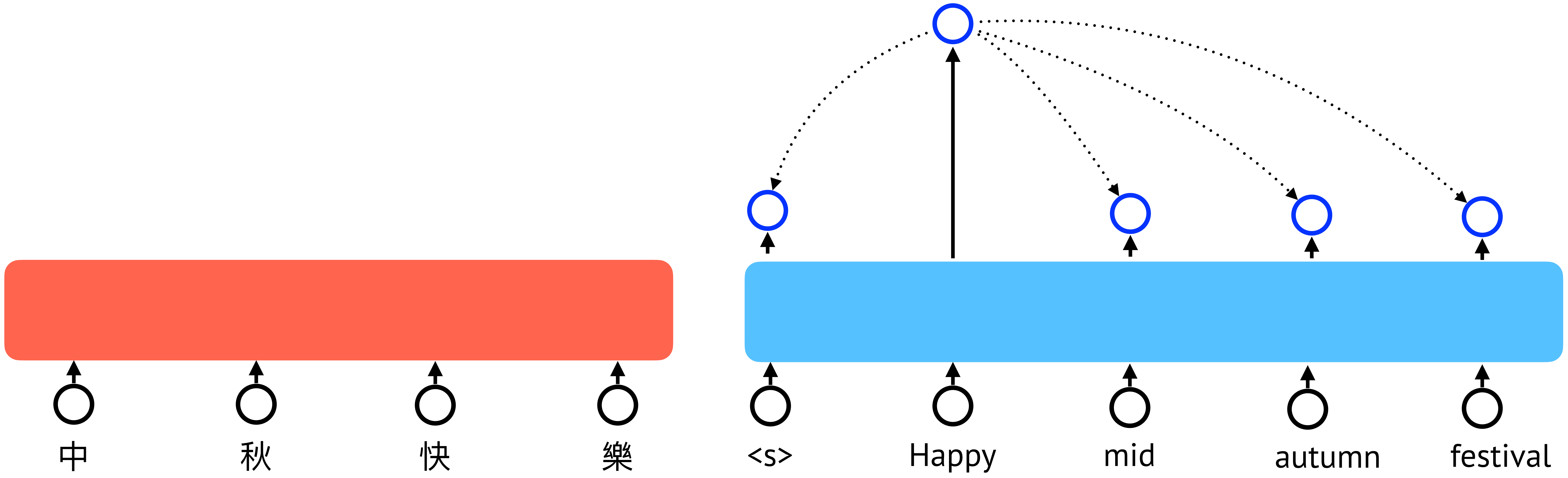


(Coda; Zheng et al, ACL 2021)

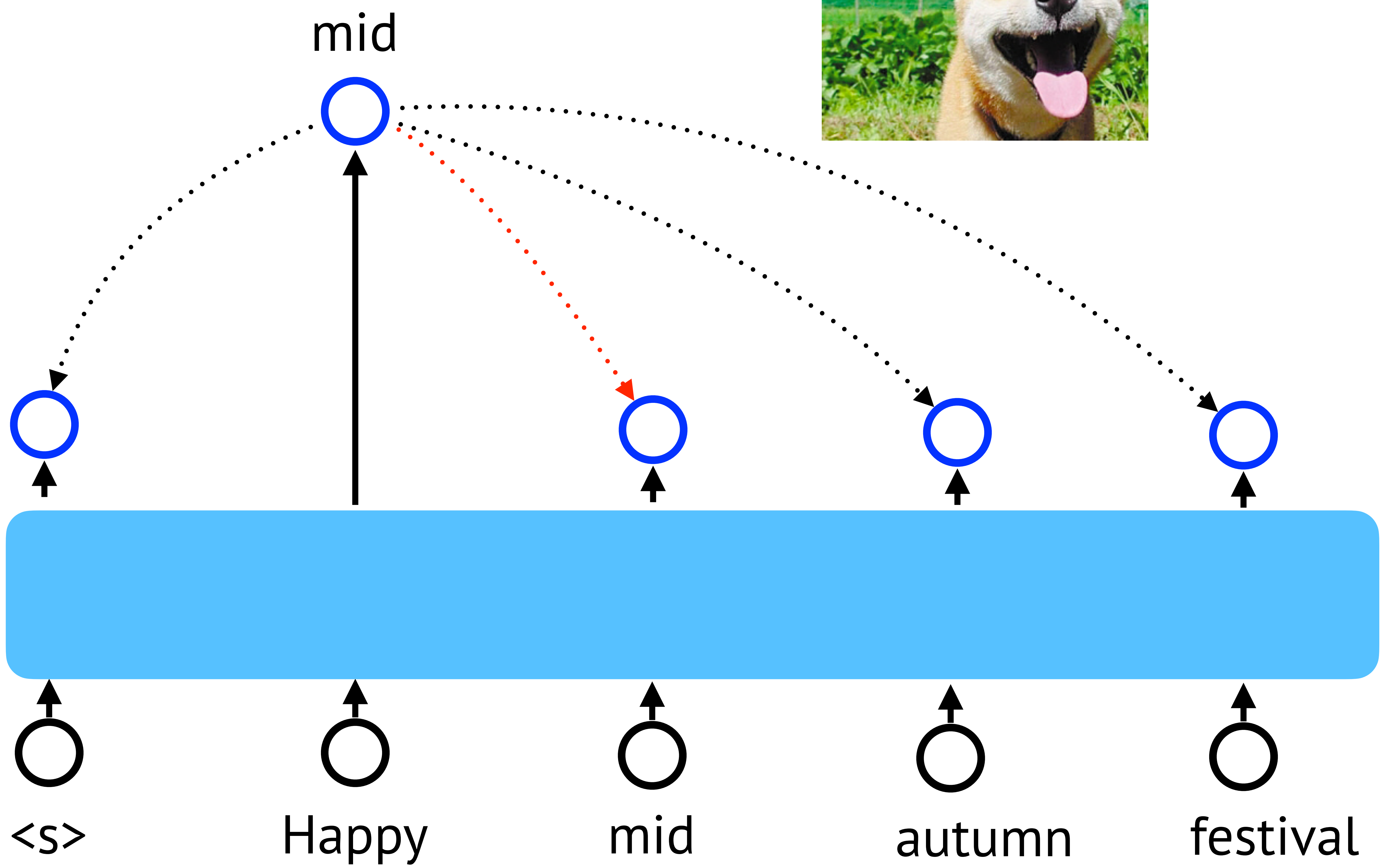
Transformer as Decoder

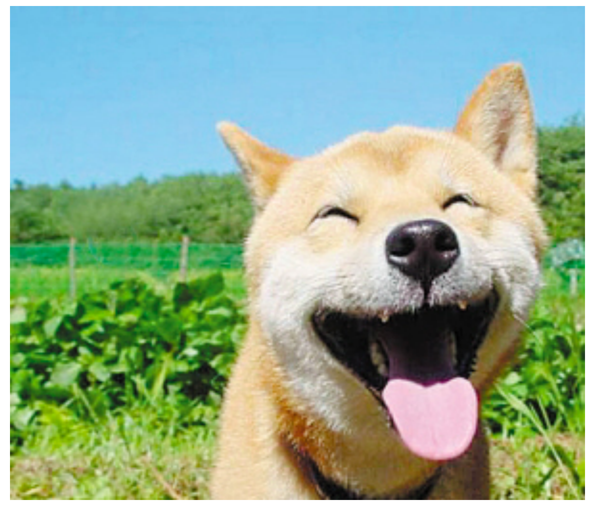


Transformer as Decoder



Transformer as Decoder



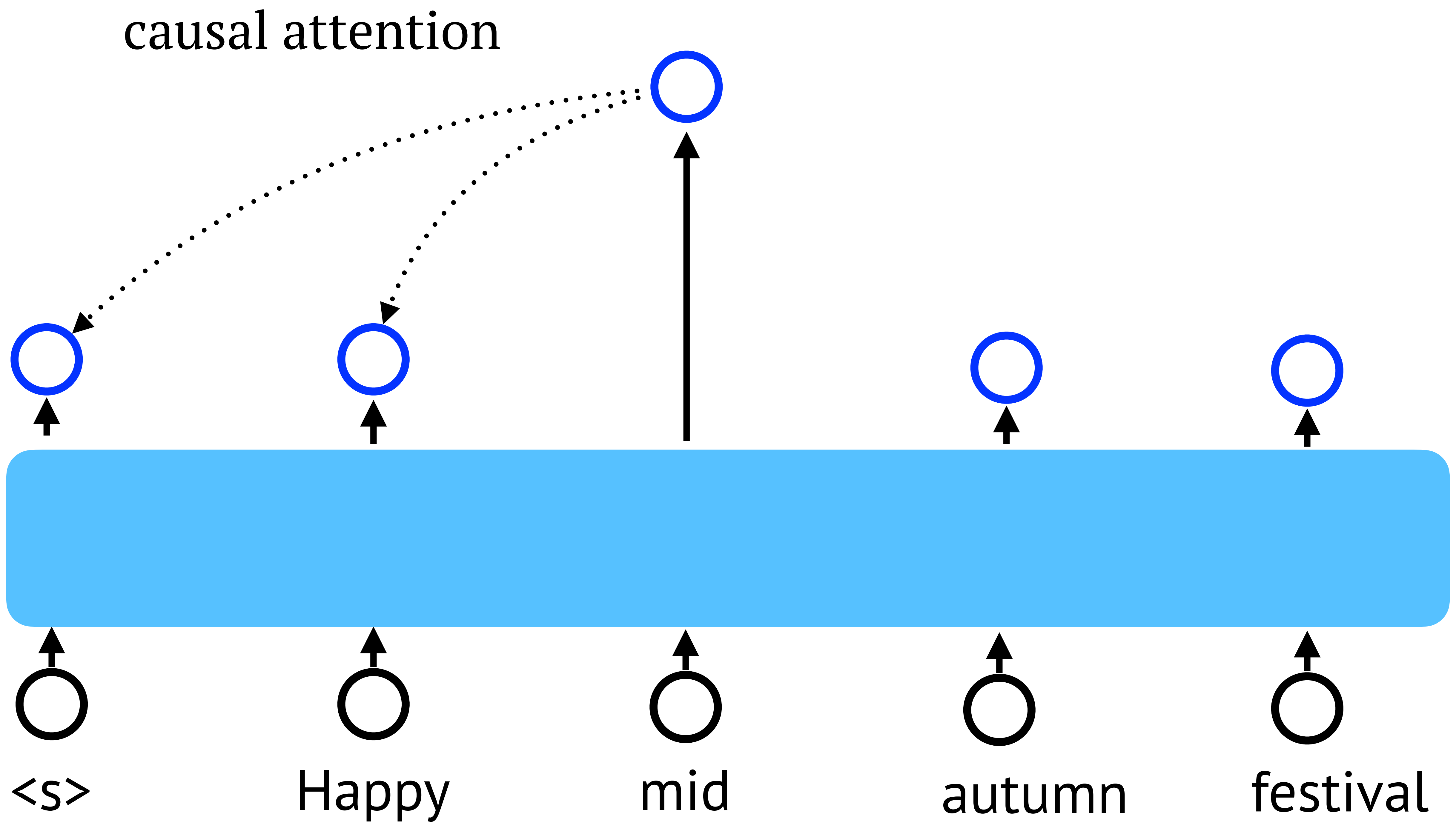


Need to prevent the attention the future words.

	Happy	mid	autumn	festival
Happy	$-\infty$	$-\infty$	$-\infty$	$-\infty$
mid		$-\infty$	$-\infty$	$-\infty$
autumn			$-\infty$	$-\infty$
festival				$-\infty$

$$e_{ij} = \begin{cases} q_i^{\top} k_j, j < i \\ -\infty, j \geq i \end{cases}$$

Transformer as Decoder

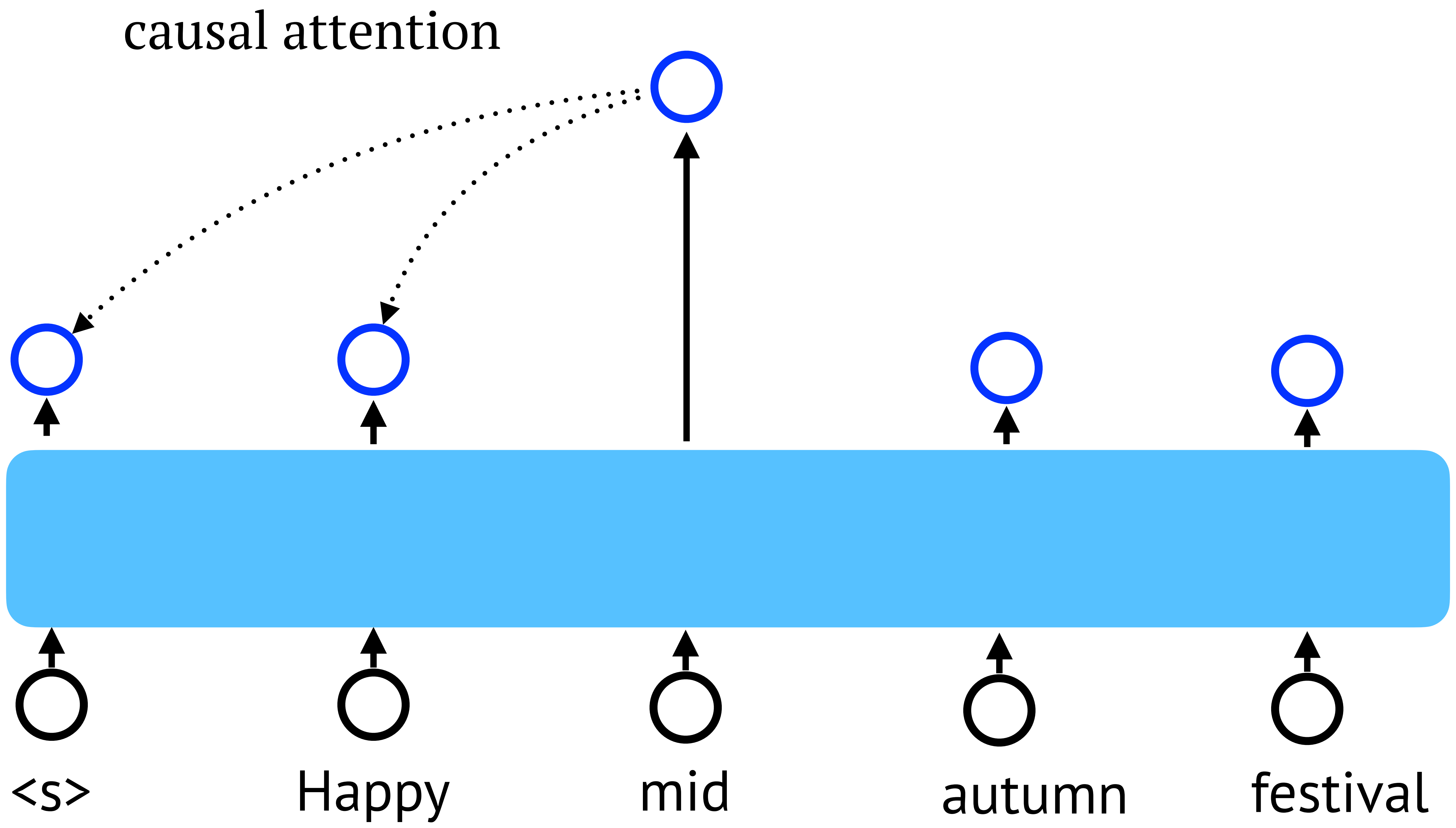


Need to prevent the attention the future words.

	Happy	mid	autumn	festival
Happy	$-\infty$	$-\infty$	$-\infty$	$-\infty$
mid		$-\infty$	$-\infty$	$-\infty$
autumn			$-\infty$	$-\infty$
festival				$-\infty$

$$e_{ij} = \begin{cases} q_i^\top k_j, j < i \\ -\infty, j \geq i \end{cases}$$

Transformer as Decoder



Need to prevent the attention the future words.

	Happy	mid	autumn	festival
Happy	$-\infty$	$-\infty$	$-\infty$	$-\infty$
mid		$-\infty$	$-\infty$	$-\infty$
autumn			$-\infty$	$-\infty$
festival				$-\infty$

$$e_{ij} = \begin{cases} q_i^\top k_j, j < i \\ -\infty, j \geq i \end{cases}$$