Applied Deep Learning

Transformer

October 13th, 2022 http://adl.miulab.tw

National Taiwan University
國立臺灣大學

2

Sequence Encoding

Basic Attention

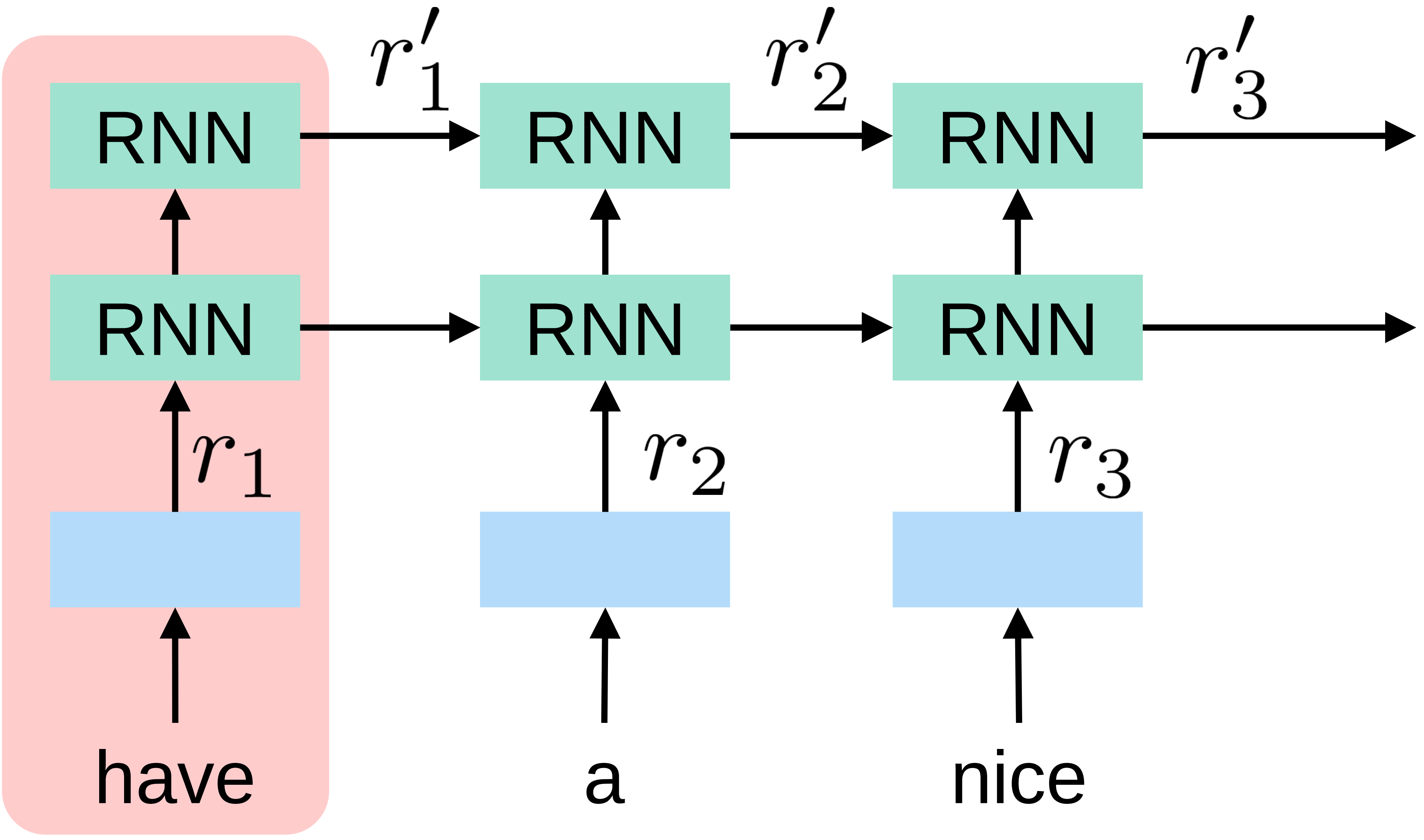
Representations of Variable Length Data

- Input: word sequence, image pixels, audio signal, click logs
- Property: continuity, temporal, importance distribution
- Example
  - ✓ Basic combination: average, sum
  - ✓ Neural combination: network architectures should consider input domain properties
    - – CNN (convolutional neural network)
    - – RNN (recurrent neural network): temporal information

Network architectures should consider the input domain properties

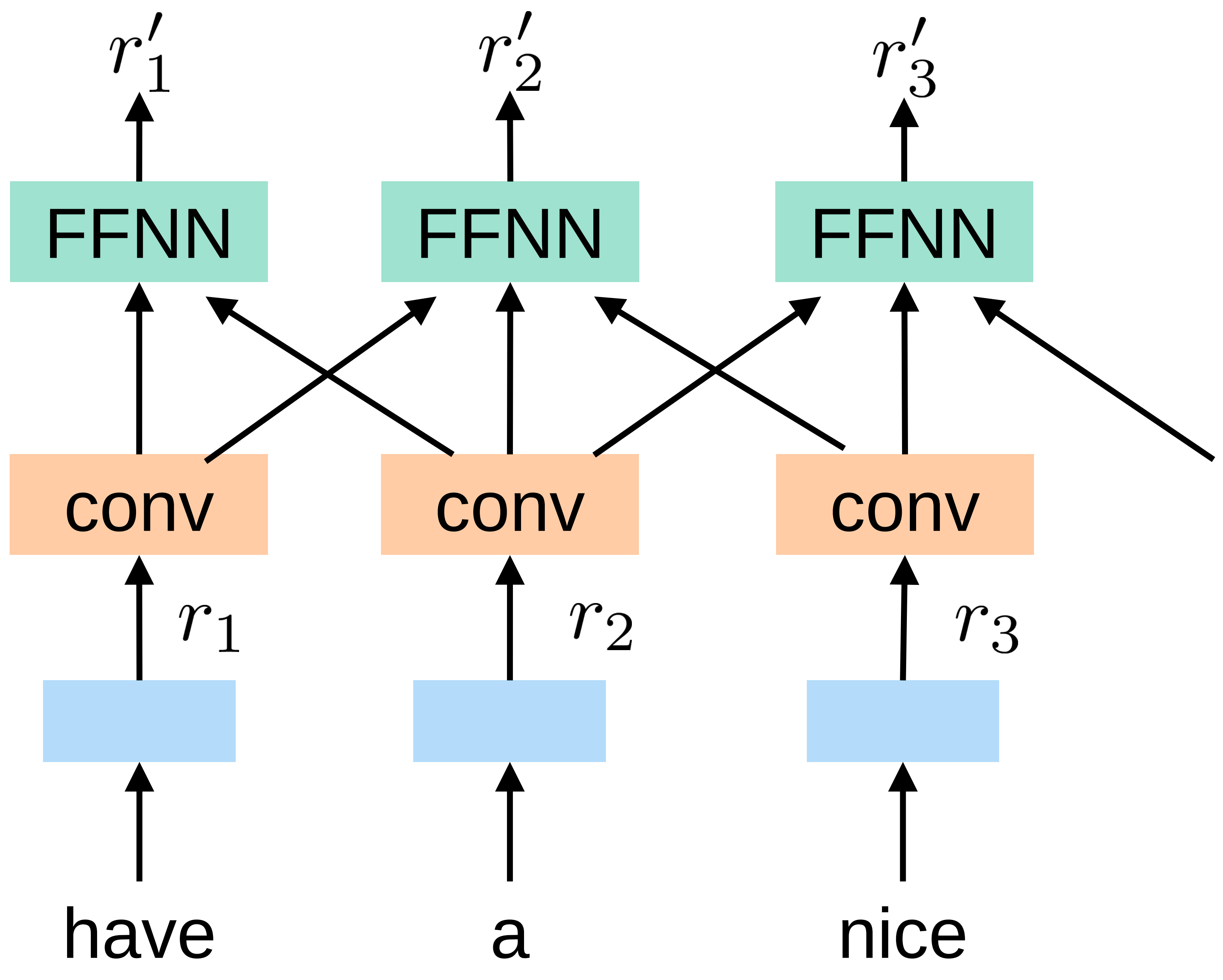
4 Recurrent Neural Networks

- Learning variable-length representations
  - ✓ Fit for sentences and sequences of values
- Sequential computation makes parallelization difficult
- No explicit modeling of long and short range dependencies



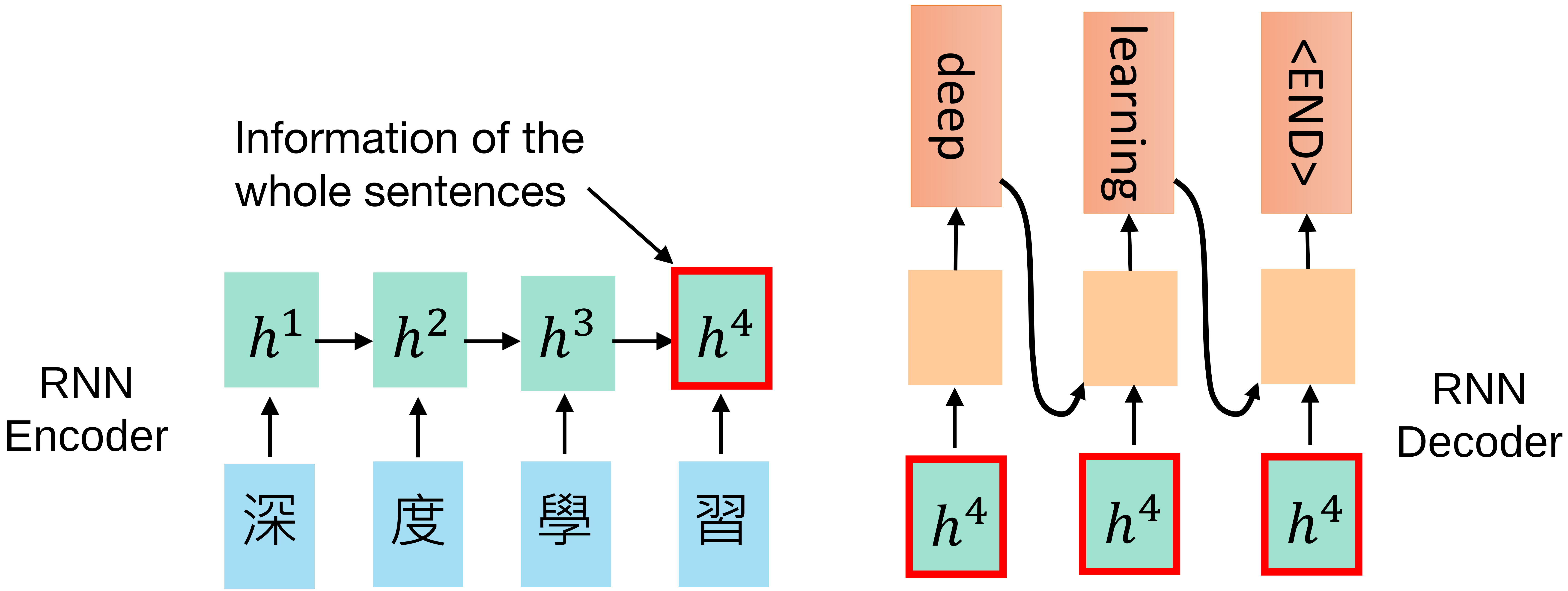
5 Convolutional Neural Networks

- Easy to parallelize
- Exploit local dependencies
  - ✓ **Long-distance** dependencies require many layers

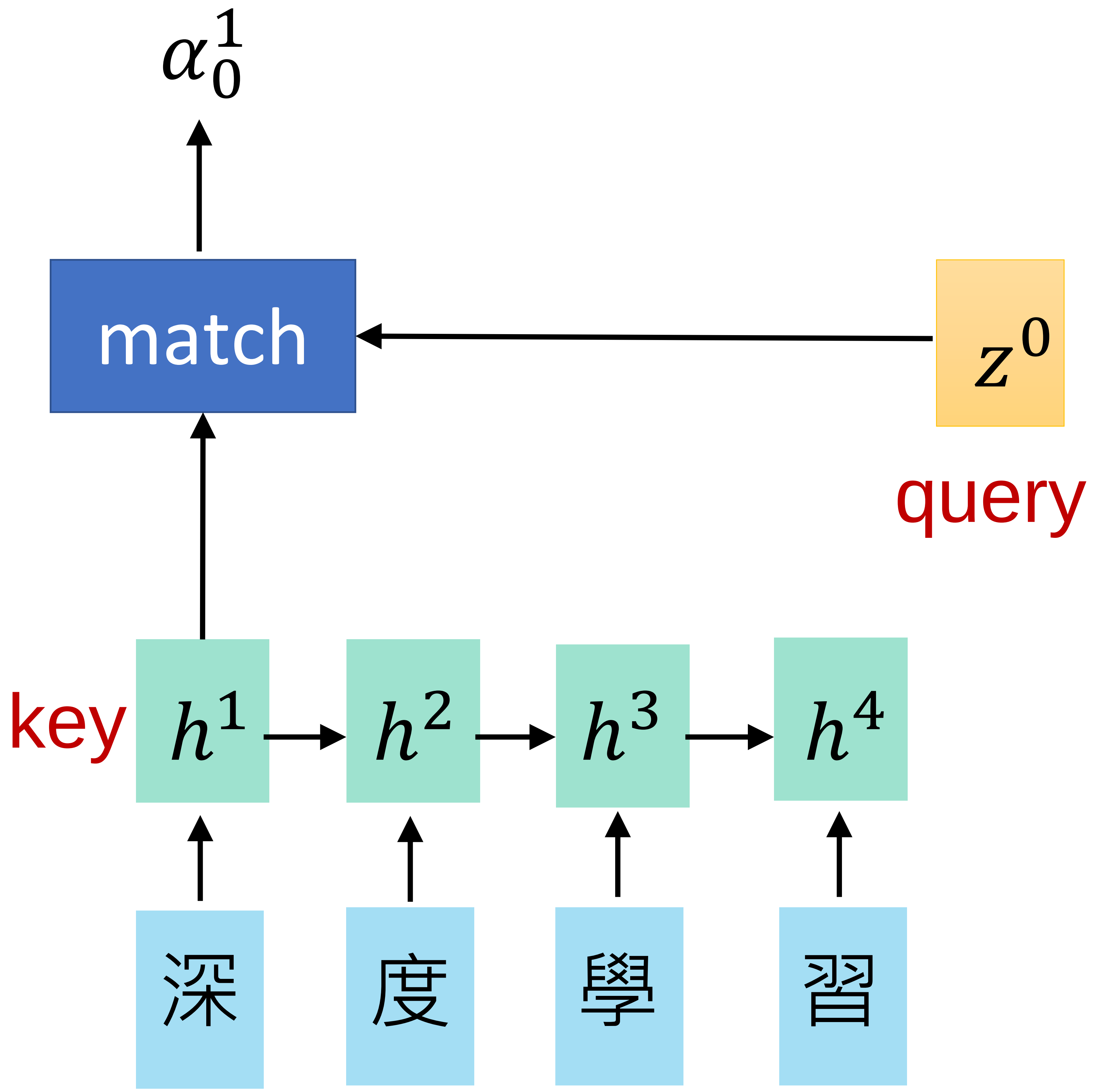


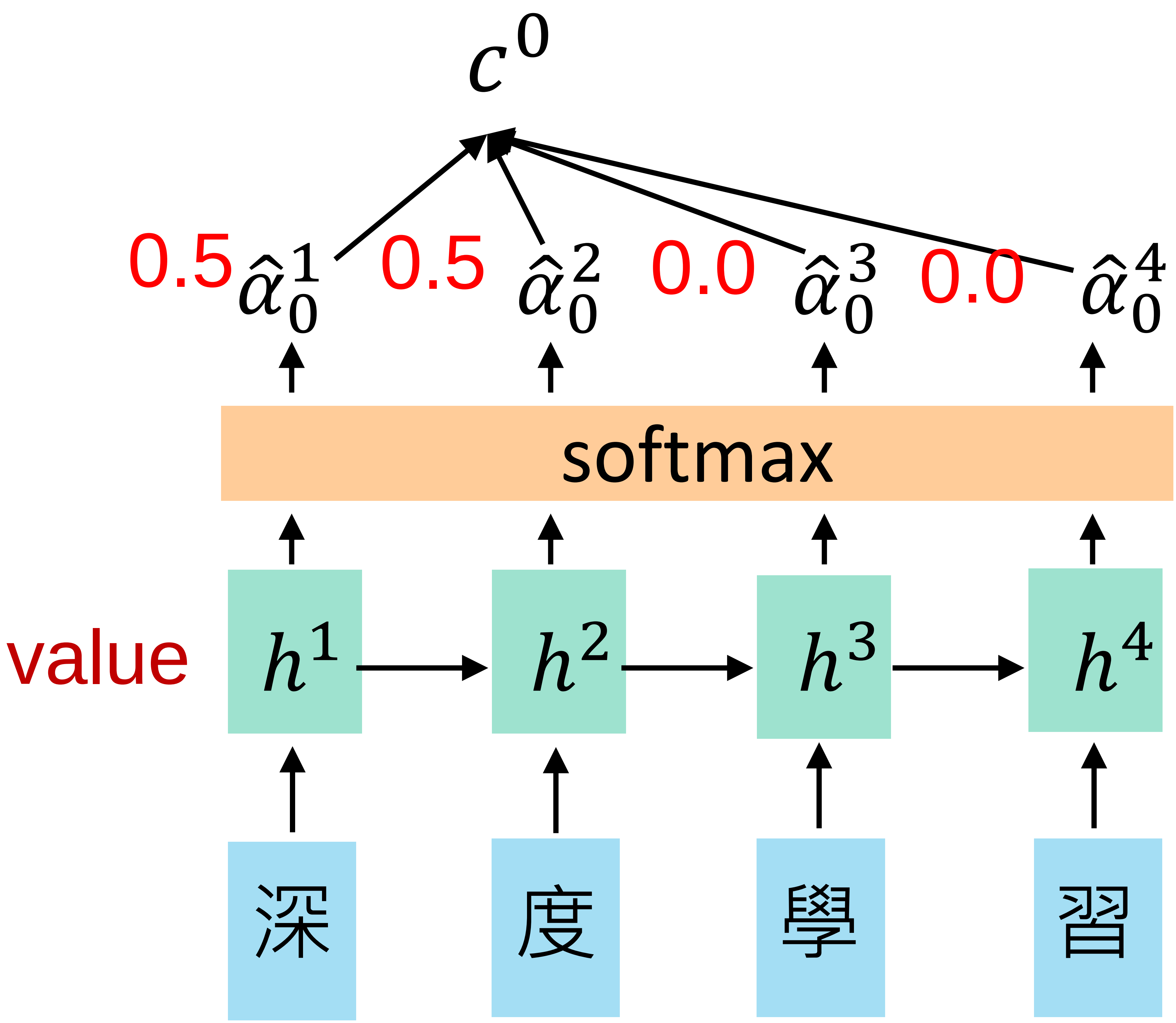
6 Attention

- Encoder-decoder model is important in NMT
- RNNs need **attention mechanism** to handle long dependencies
- Attention allows us to access any state



Basic Attention





Dot-Product Attention

- Input: a query q and a set of key-value (k-v) pairs to an output
- Output: weighted sum of values

Inner product of
query and corresponding key

$$A(q, K, V) = \sum_i \boxed{\frac{\exp(q \cdot k_i)}{\sum_j \exp(q \cdot k_j)}} v_i$$

- ✓ Query q is a d_k-dim vector
- ✓ Key k is a d_k-dim vector
- ✓ Value v is a d_v-dim vector

Dot-Product Attention in Matrix

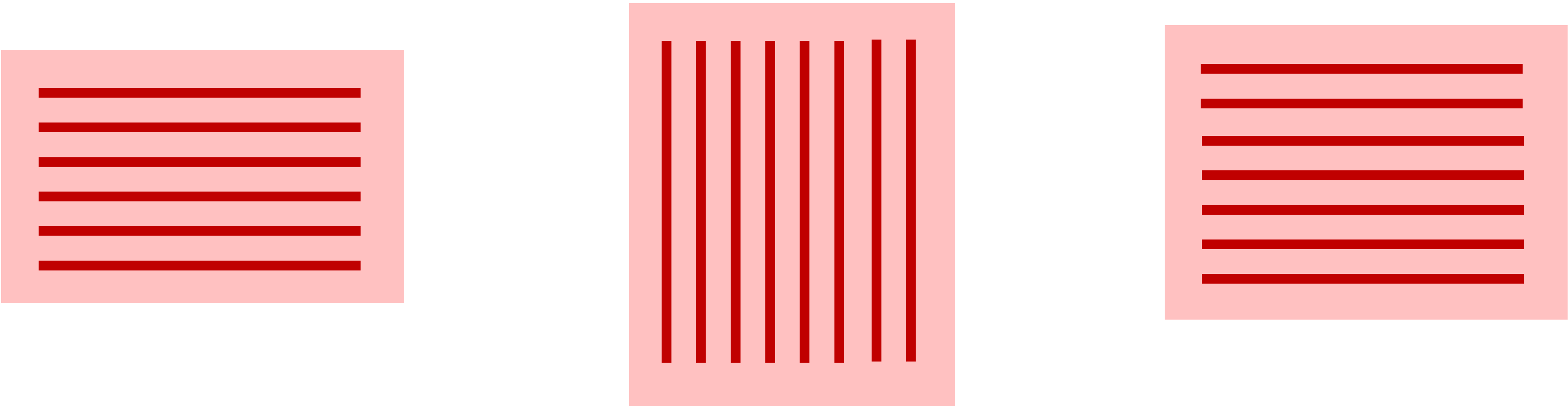
- Input: *multiple* queries q and a set of key-value (k-v) pairs to an output
- Output: a set of weighted sum of values

$$A(q, K, V) = \sum_i \frac{\exp(q \cdot k_i)}{\sum_j \exp(q \cdot k_j)} v_i$$

$$A(Q, K, V) = \mathrm{softmax}(QK^T)V$$

$$[|Q| \times d_k] \times [d_k \times |K|] \times [|K| \times d_v]$$

softmax
row-wise



$$= [|Q| \times d_v]$$

Sequence Encoding
Self-Attention

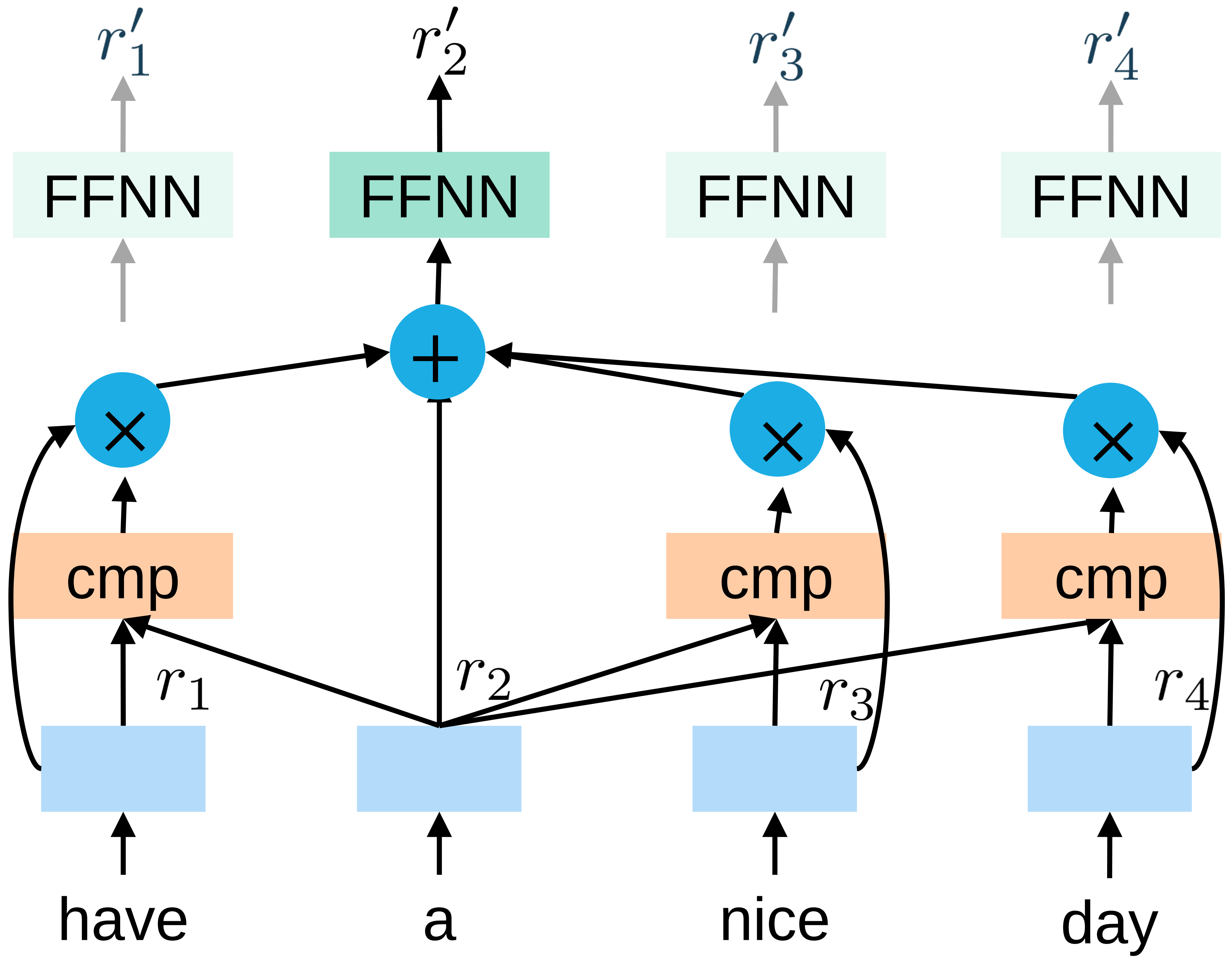
Attention

- Encoder-decoder model is important in NMT
- RNNs need **attention mechanism** to handle long dependencies
- Attention allows us to access any state

Using attention to replace recurrence architectures

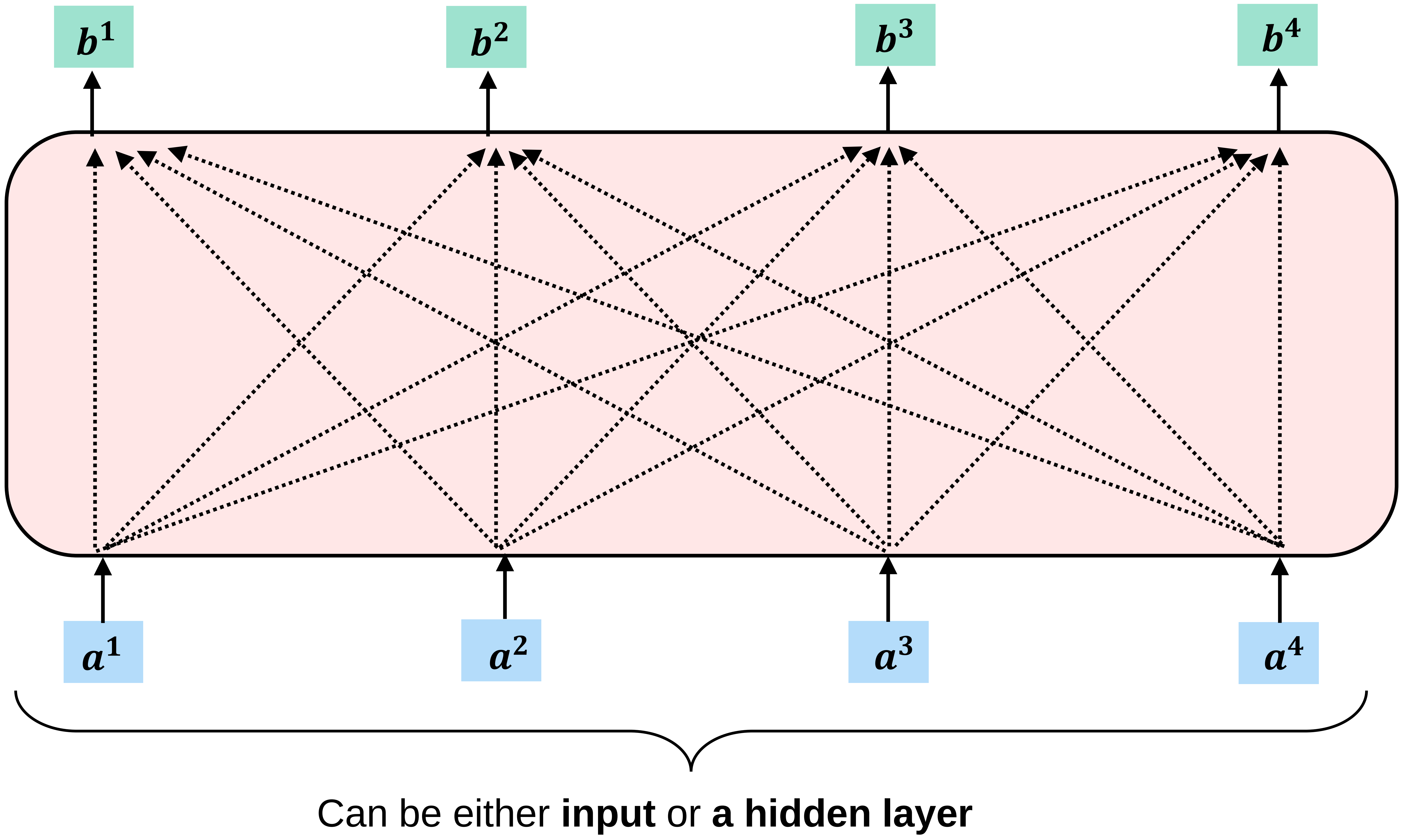
Self-Attention

- Constant "path length" between two positions
- Easy to parallelize



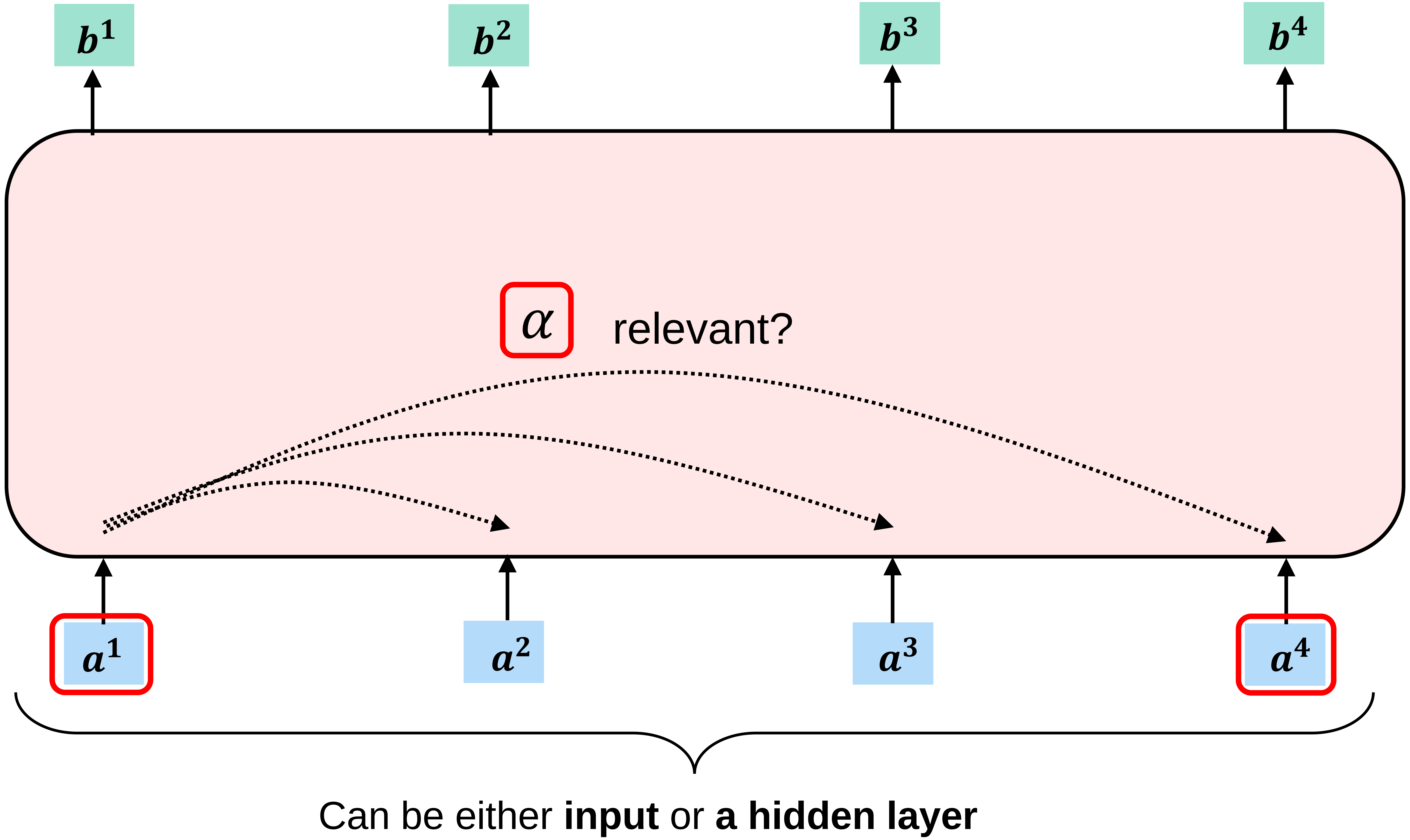
Self-Attention

- Constant “path length” between two positions
- Easy to parallelize



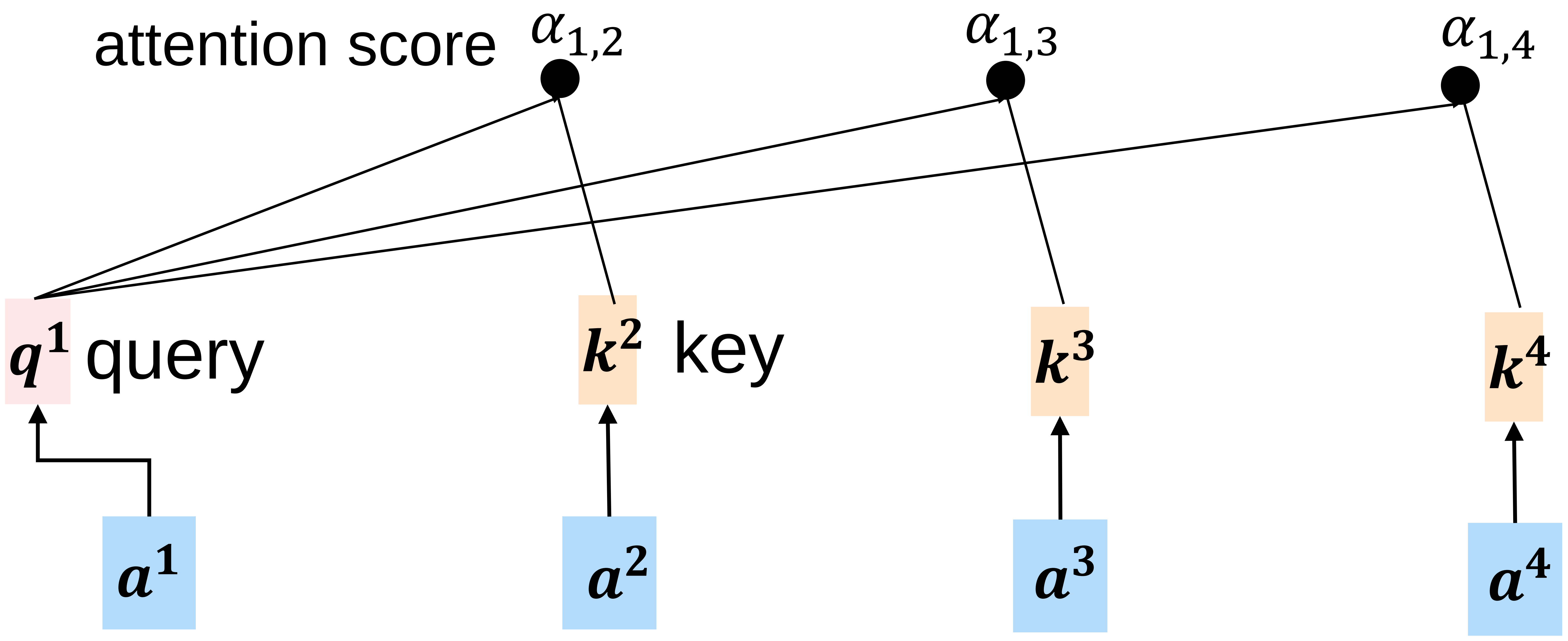
Self-Attention

- Constant “path length” between two positions
- Easy to parallelize



Self-Attention

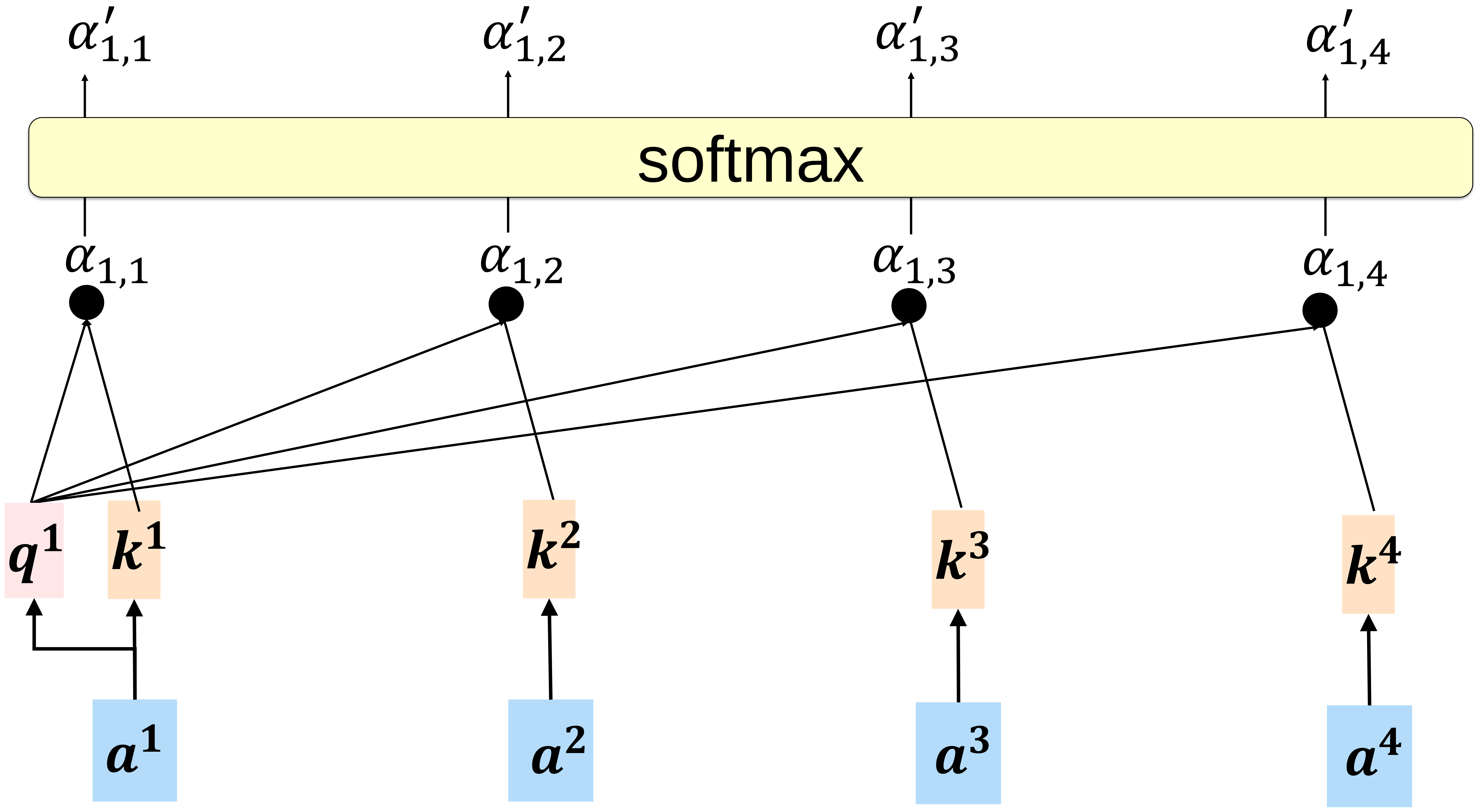
$\alpha_{1,2} = \boldsymbol{q}^1 \cdot \boldsymbol{k}^2$ $\quad$ $\alpha_{1,3} = \boldsymbol{q}^1 \cdot \boldsymbol{k}^3$ $\quad$ $\alpha_{1,4} = \boldsymbol{q}^1 \cdot \boldsymbol{k}^4$



$\boldsymbol{q}^1 = W^q \boldsymbol{a}^1$ $\quad$ $\boldsymbol{k}^2 = W^k \boldsymbol{a}^2$ $\quad$ $\boldsymbol{k}^3 = W^k \boldsymbol{a}^3$ $\quad$ $\boldsymbol{k}^4 = W^k \boldsymbol{a}^4$

Self-Attention

$$\alpha'_{1,i} = e^{\alpha_{1,i}} / \sum_j e^{\alpha_{1,j}}$$



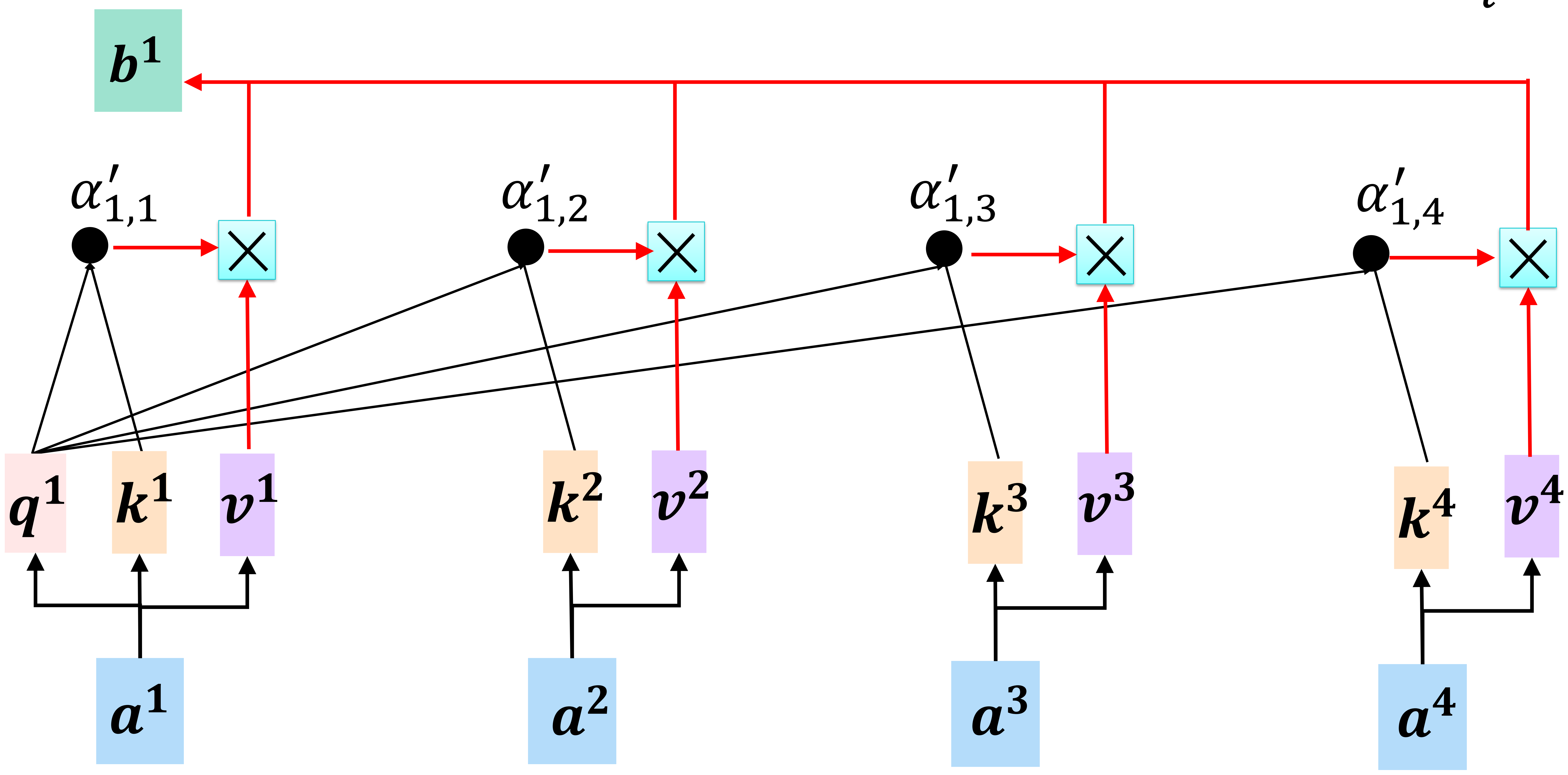
$\boldsymbol{q}^1 = W^q \boldsymbol{a}^1$ $\quad$ $\boldsymbol{k}^2 = W^k \boldsymbol{a}^2$ $\quad$ $\boldsymbol{k}^3 = W^k \boldsymbol{a}^3$ $\quad$ $\boldsymbol{k}^4 = W^k \boldsymbol{a}^4$

$\boldsymbol{k}^1 = W^k \boldsymbol{a}^1$

Self-Attention

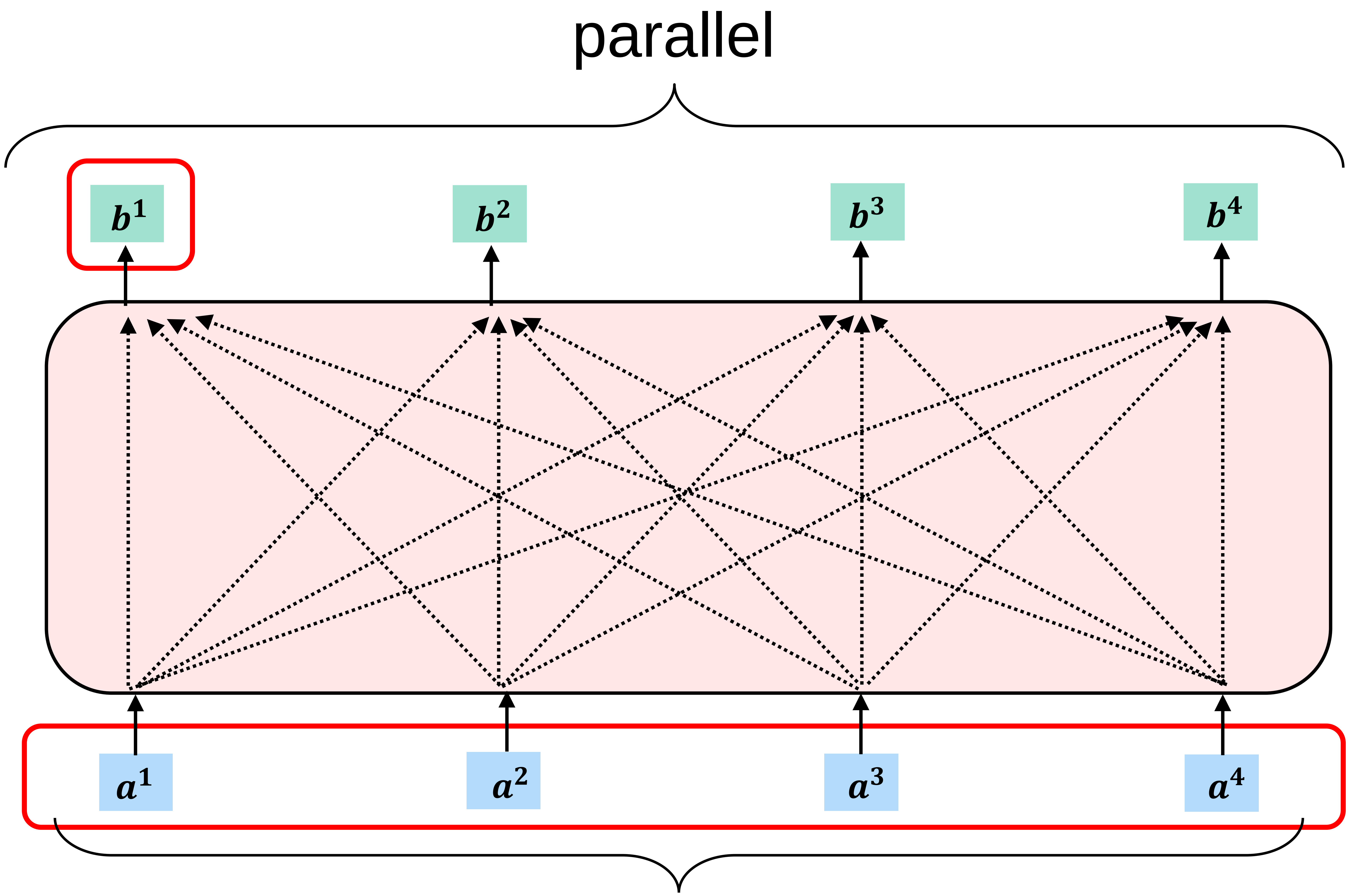
extract information based on attention scores

$$\boldsymbol{b}^1 = \sum_i \alpha'_{1,i} \boldsymbol{v}^i$$



$\boldsymbol{v}^1 = W^v \boldsymbol{a}^1$ $\boldsymbol{v}^2 = W^v \boldsymbol{a}^2$ $\boldsymbol{v}^3 = W^v \boldsymbol{a}^3$ $\boldsymbol{v}^4 = W^v \boldsymbol{a}^4$

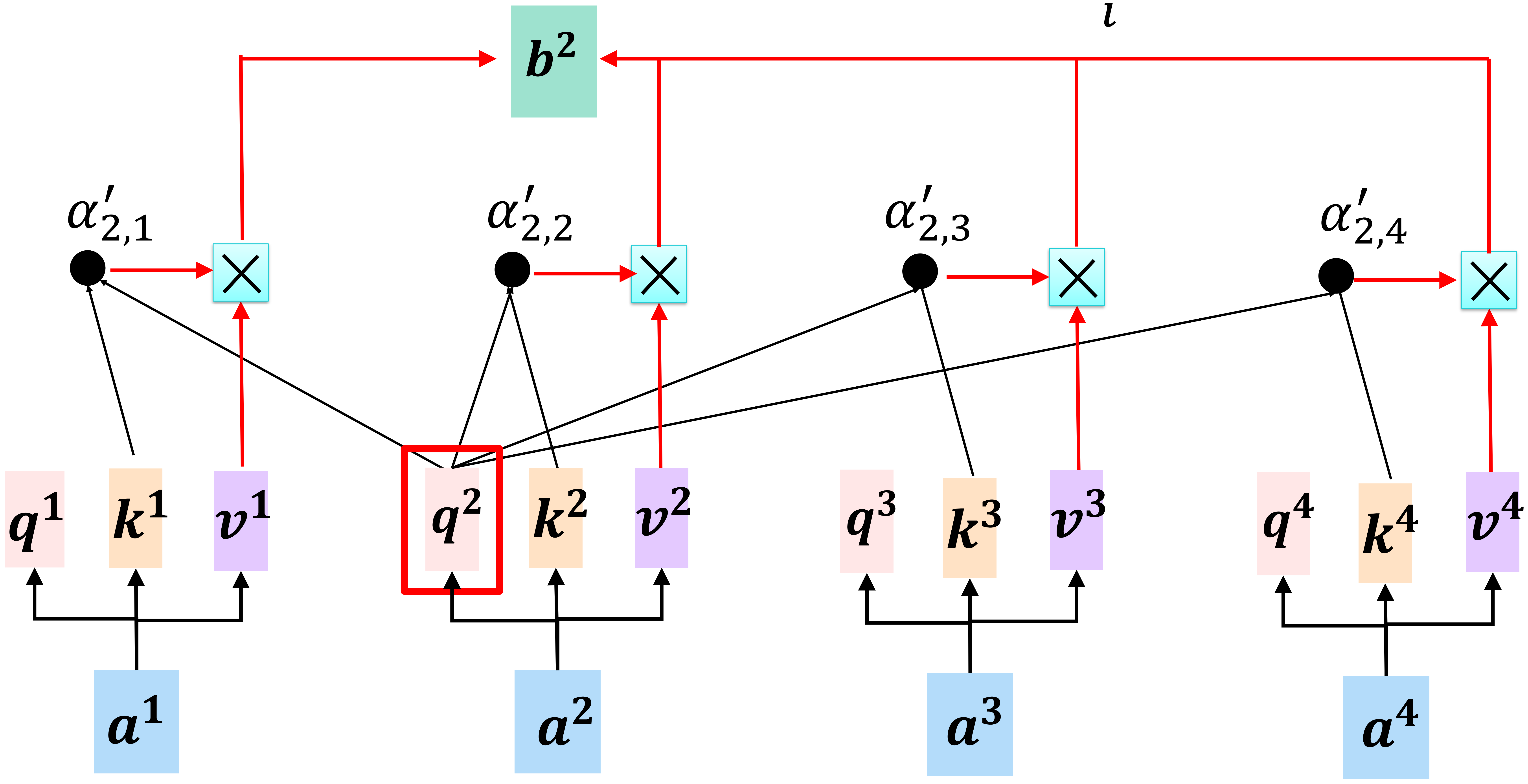
Self-Attention



Can be either **input** or **a hidden layer**

Self-Attention

$$b^2 = \sum_i \alpha'_{2,i} v^i$$

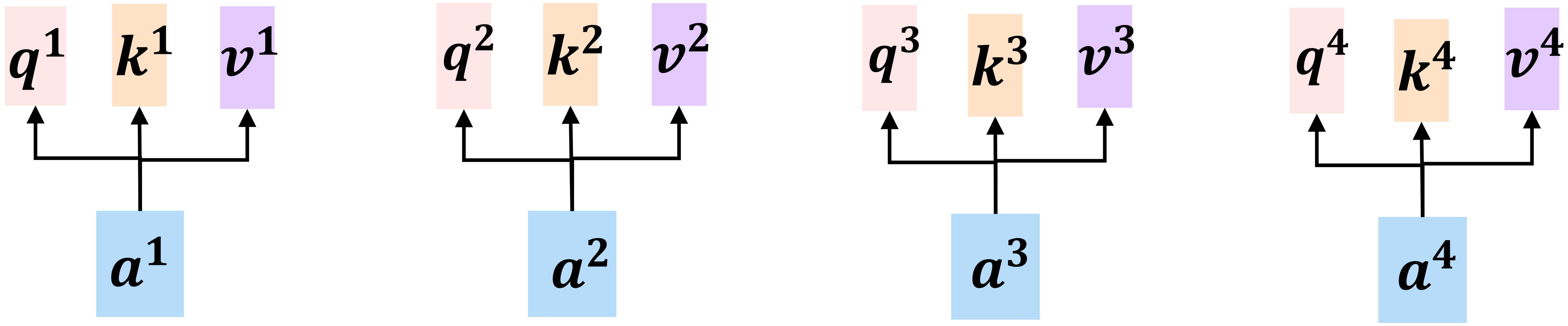


Self-Attention

$$q^i = W^q a^i \qquad \underbrace{q^1\ q^2\ q^3\ q^4}_{Q} = W^q\ \underbrace{a^1\ a^2\ a^3\ a^4}_{I}$$

$$k^i = W^k a^i \qquad \underbrace{k^1\ k^2\ k^3\ k^4}_{K} = W^k\ \underbrace{a^1\ a^2\ a^3\ a^4}_{I}$$

$$v^i = W^v a^i \qquad \underbrace{v^1\ v^2\ v^3\ v^4}_{V} = W^v\ \underbrace{a^1\ a^2\ a^3\ a^4}_{I}$$

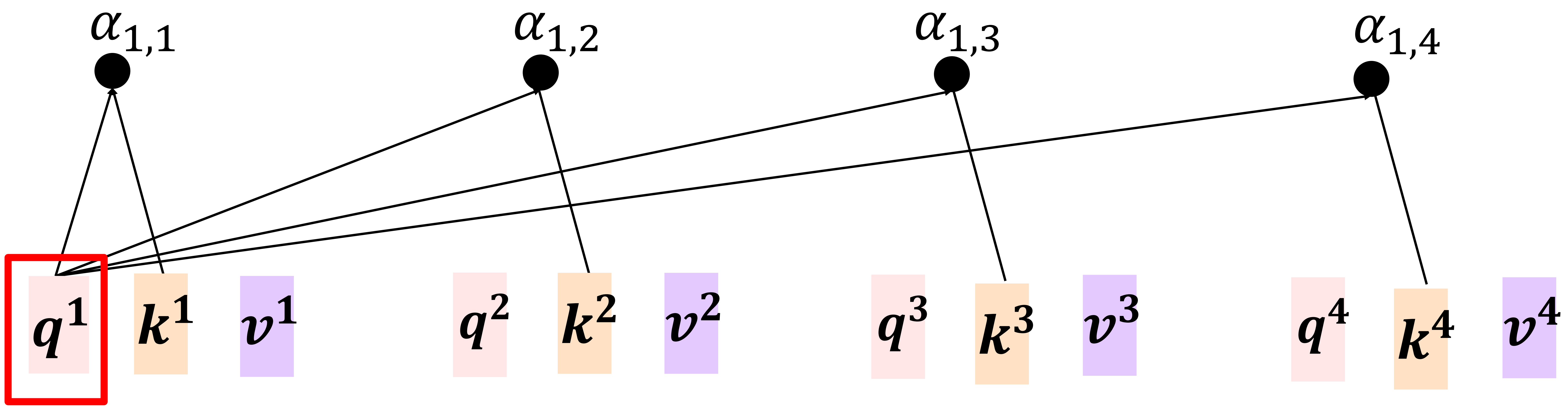


Self-Attention

$$\alpha_{1,1} = \boldsymbol{k}^1 \; \boldsymbol{q}^1 \qquad \alpha_{1,2} = \boldsymbol{k}^2 \; \boldsymbol{q}^1$$

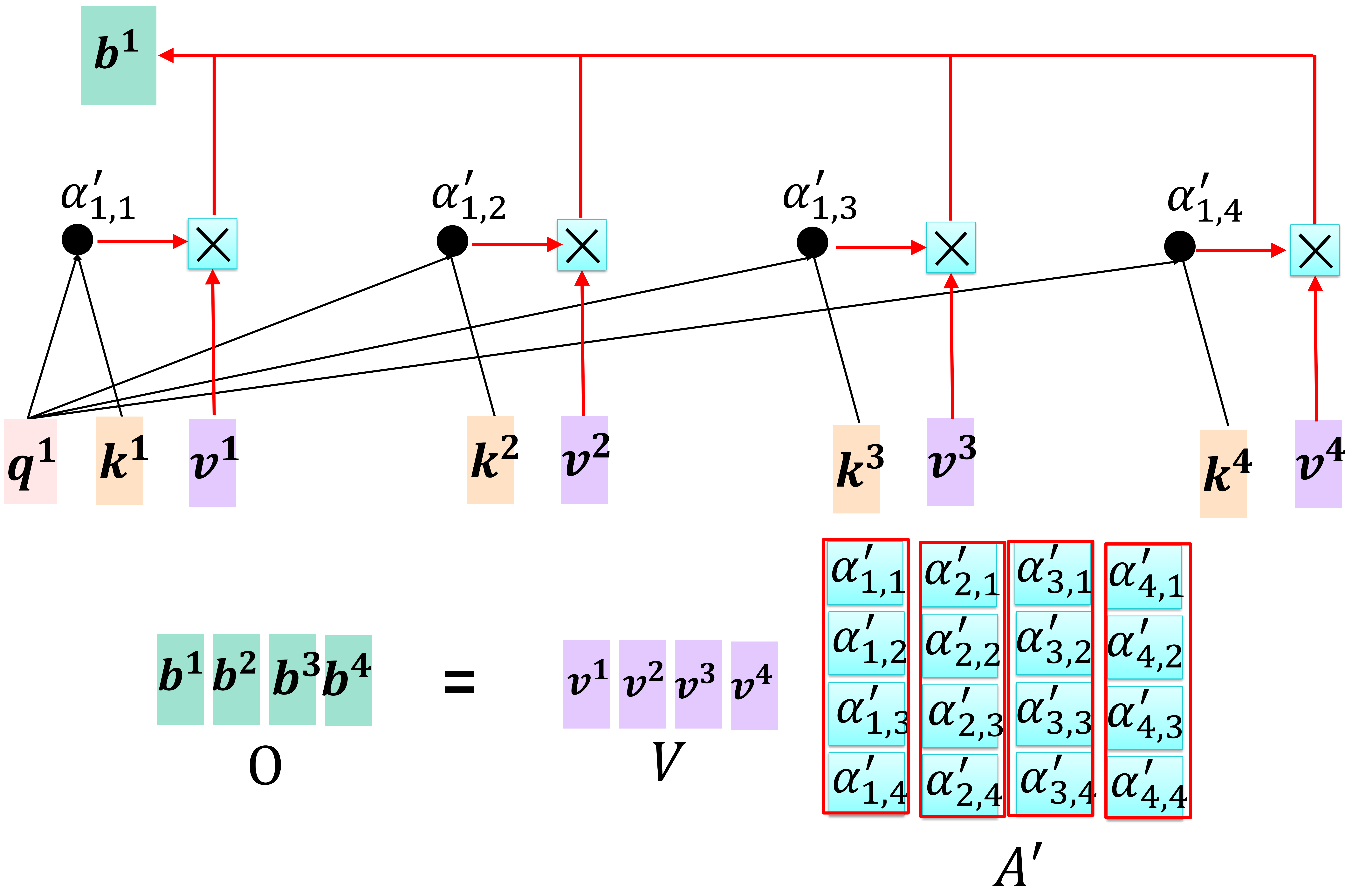
$$\alpha_{1,3} = \boldsymbol{k}^3 \; \boldsymbol{q}^1 \qquad \alpha_{1,4} = \boldsymbol{k}^4 \; \boldsymbol{q}^1$$

$$\begin{bmatrix} \alpha_{1,1} \\ \alpha_{1,2} \\ \alpha_{1,3} \\ \alpha_{1,4} \end{bmatrix} = \begin{bmatrix} \boldsymbol{k}^1 \\ \boldsymbol{k}^2 \\ \boldsymbol{k}^3 \\ \boldsymbol{k}^4 \end{bmatrix} \boldsymbol{q}^1$$



$$\underbrace{\begin{bmatrix} \alpha'_{1,1} & \alpha'_{2,1} & \alpha'_{3,1} & \alpha'_{4,1} \\ \alpha'_{1,2} & \alpha'_{2,2} & \alpha'_{3,2} & \alpha'_{4,2} \\ \alpha'_{1,3} & \alpha'_{2,3} & \alpha'_{3,3} & \alpha'_{4,3} \\ \alpha'_{1,4} & \alpha'_{2,4} & \alpha'_{3,4} & \alpha'_{4,4} \end{bmatrix}}_{A'} \Leftarrow \underbrace{\begin{bmatrix} \alpha_{1,1} & \alpha_{2,1} & \alpha_{3,1} & \alpha_{4,1} \\ \alpha_{1,2} & \alpha_{2,2} & \alpha_{3,2} & \alpha_{4,2} \\ \alpha_{1,3} & \alpha_{2,3} & \alpha_{3,3} & \alpha_{4,3} \\ \alpha_{1,4} & \alpha_{2,4} & \alpha_{3,4} & \alpha_{4,4} \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} \boldsymbol{k}^1 \\ \boldsymbol{k}^2 \\ \boldsymbol{k}^3 \\ \boldsymbol{k}^4 \end{bmatrix}}_{K^T} \underbrace{\begin{bmatrix} \boldsymbol{q}^1 & \boldsymbol{q}^2 & \boldsymbol{q}^3 & \boldsymbol{q}^4 \end{bmatrix}}_{Q}$$

Self-Attention



Self-Attention

$$Q = W^q I$$

$$K = W^k I$$

$$V = W^v I$$

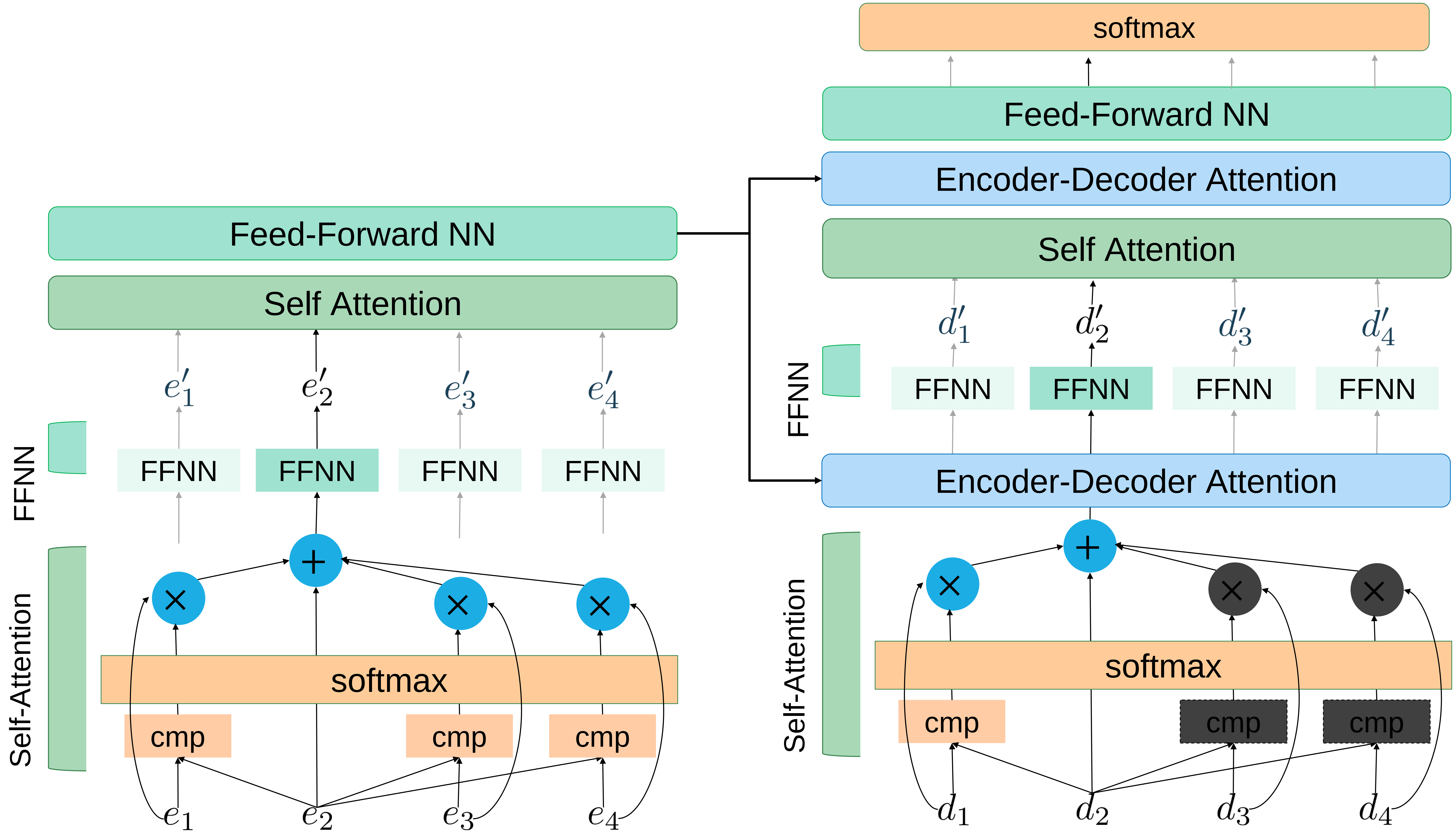
Parameters to be learned

$$A' \leftarrow A = K^T Q$$

Attention Matrix

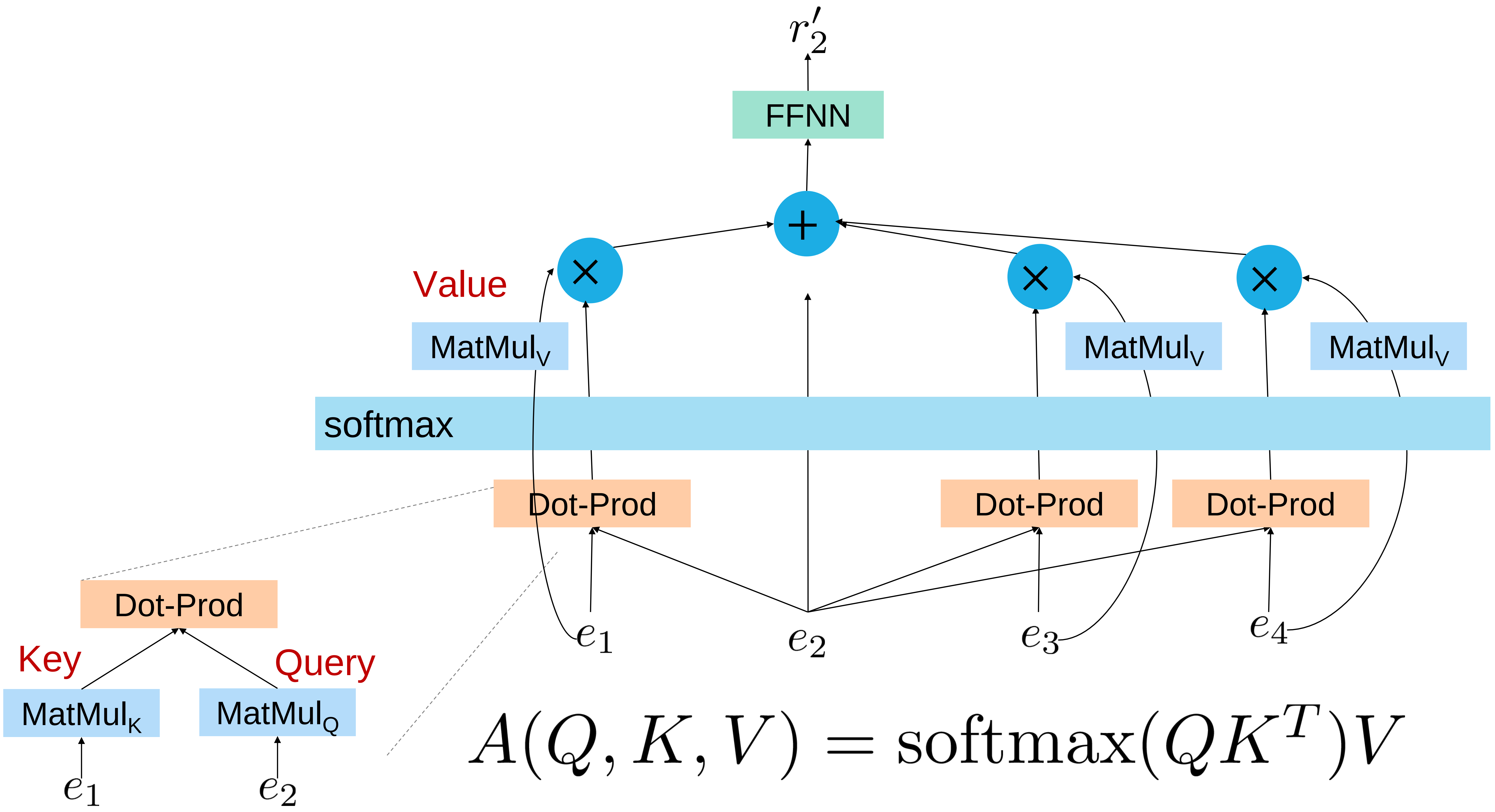
$$O = V A'$$

Transformer Idea



Vaswani et al., “Attention Is All You Need”, in *NIPS*, 2017.

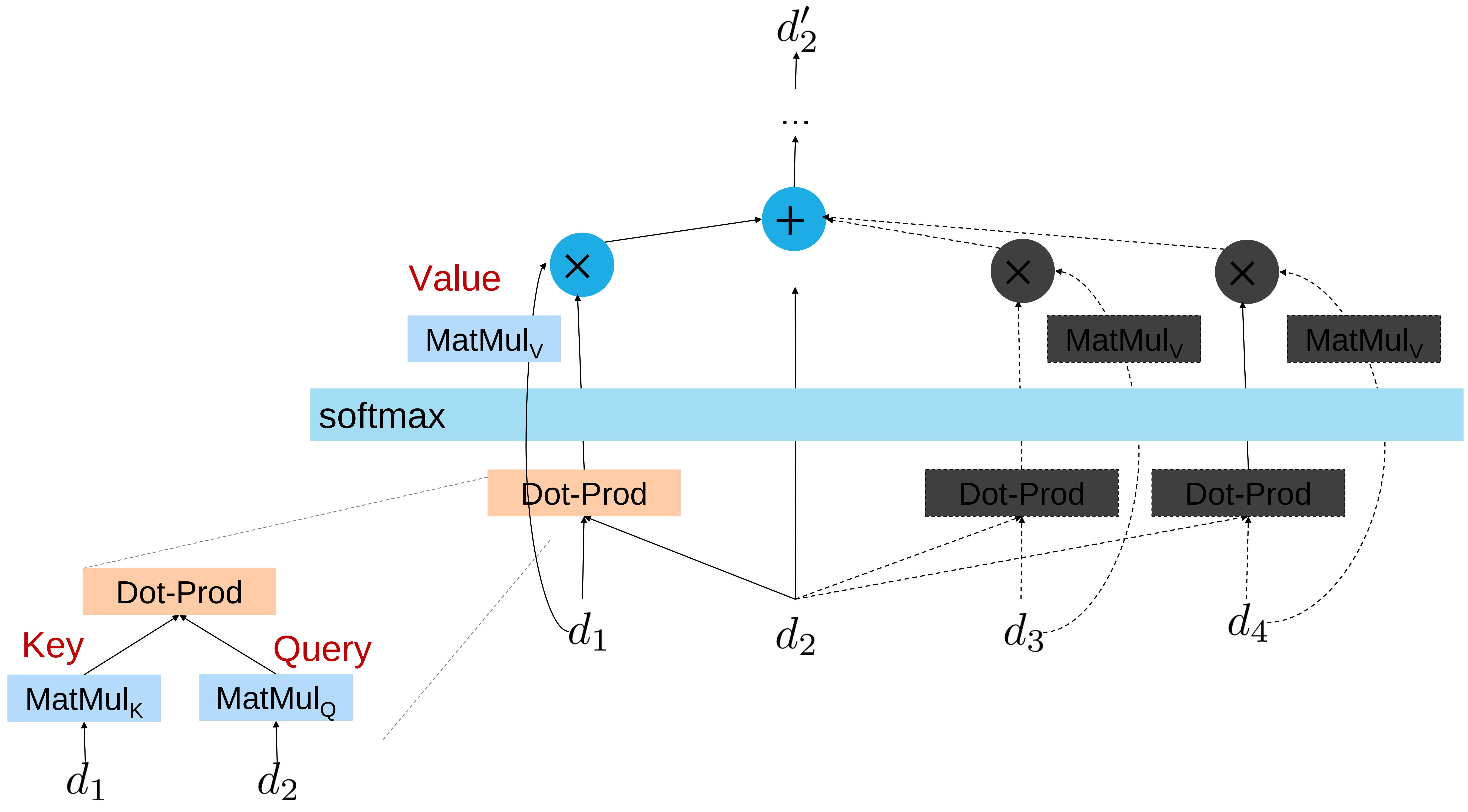
Encoder Self-Attention (Vaswani+, 2017)



$$A(Q,K,V) = \mathrm{softmax}(QK^T)V$$

Vaswani et al., “Attention Is All You Need”, in *NIPS*, 2017.

Decoder Self-Attention (Vaswani+, 2017)



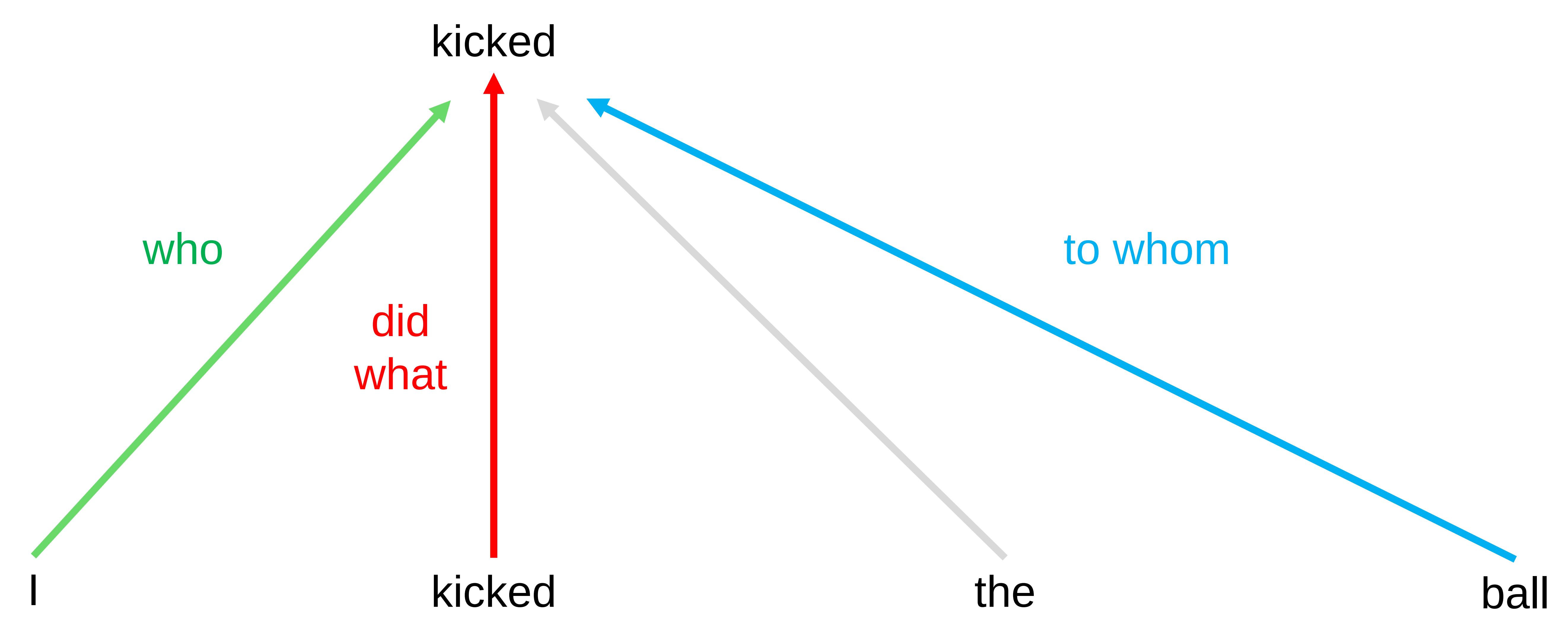
Vaswani et al., "Attention Is All You Need", in *NIPS*, 2017.

27

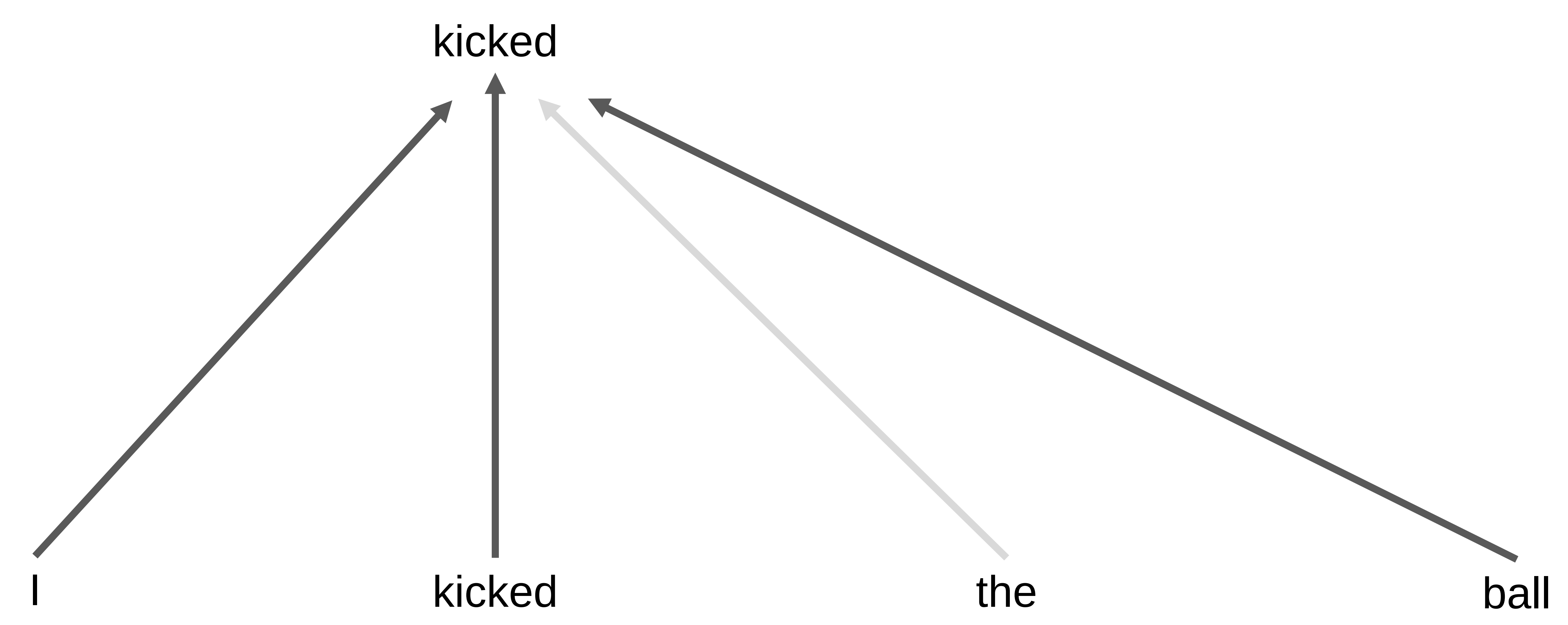
Sequence Encoding

Multi-Head Attention

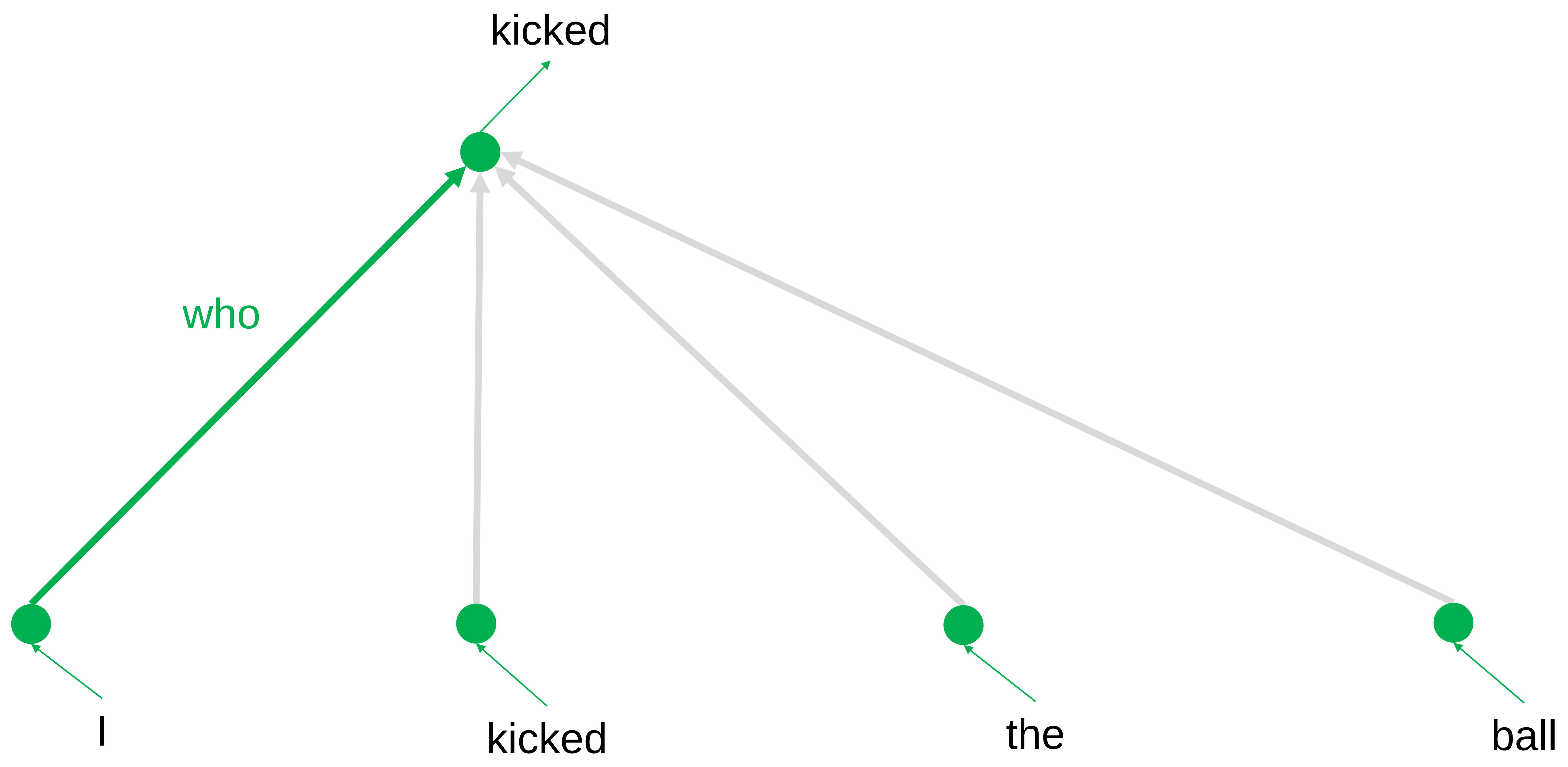
Convolutions



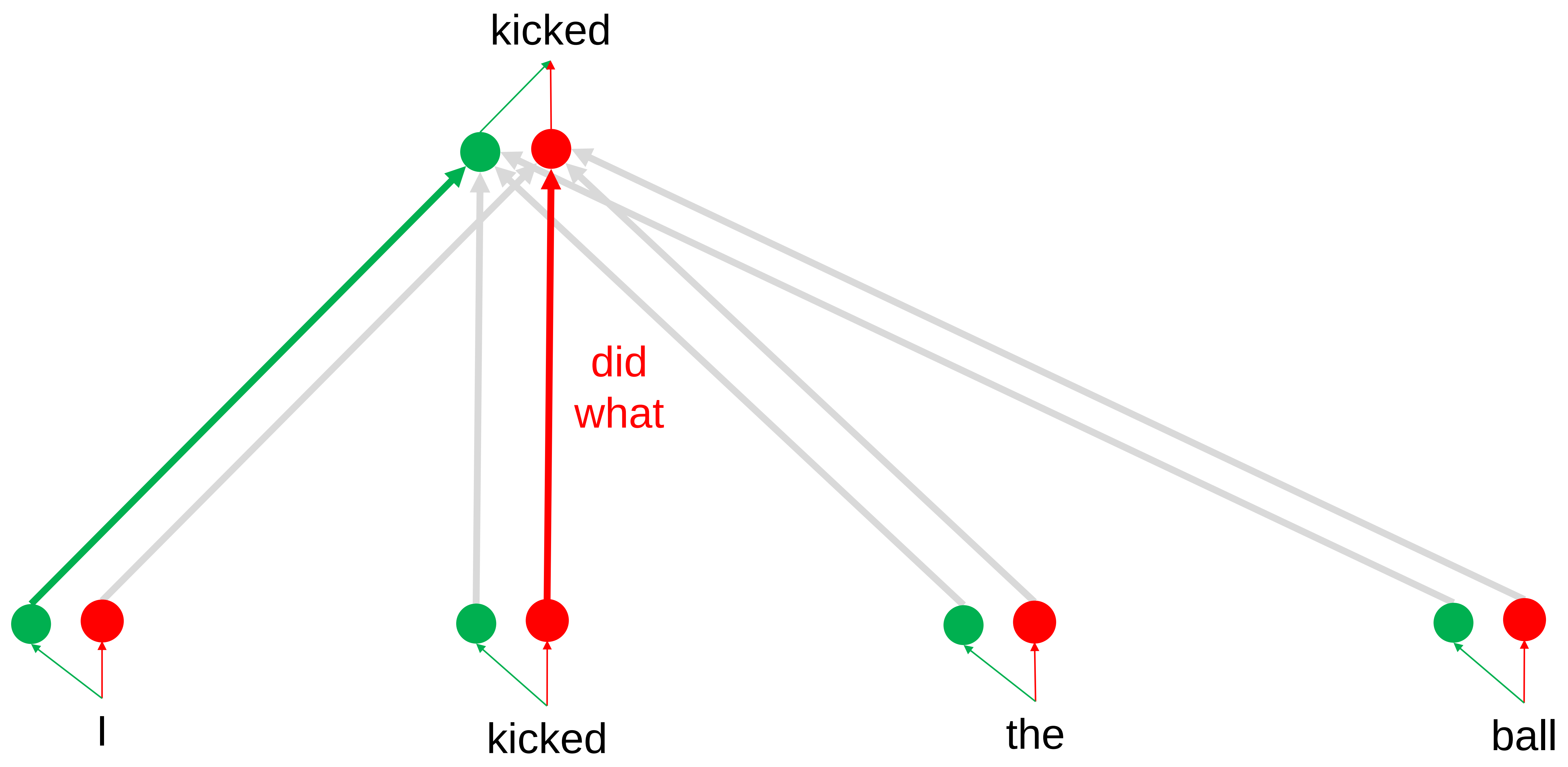
Self-Attention



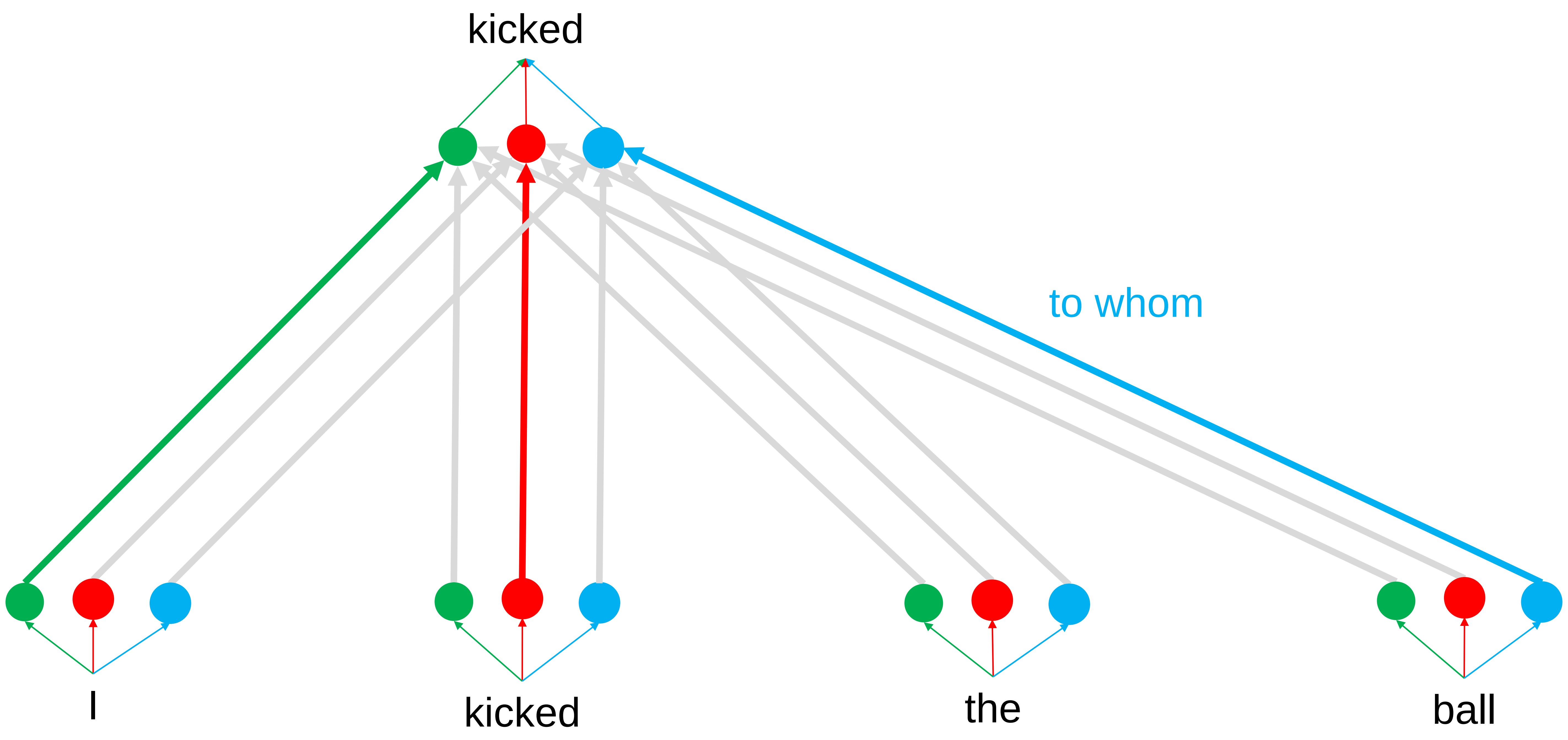
Attention Head: who



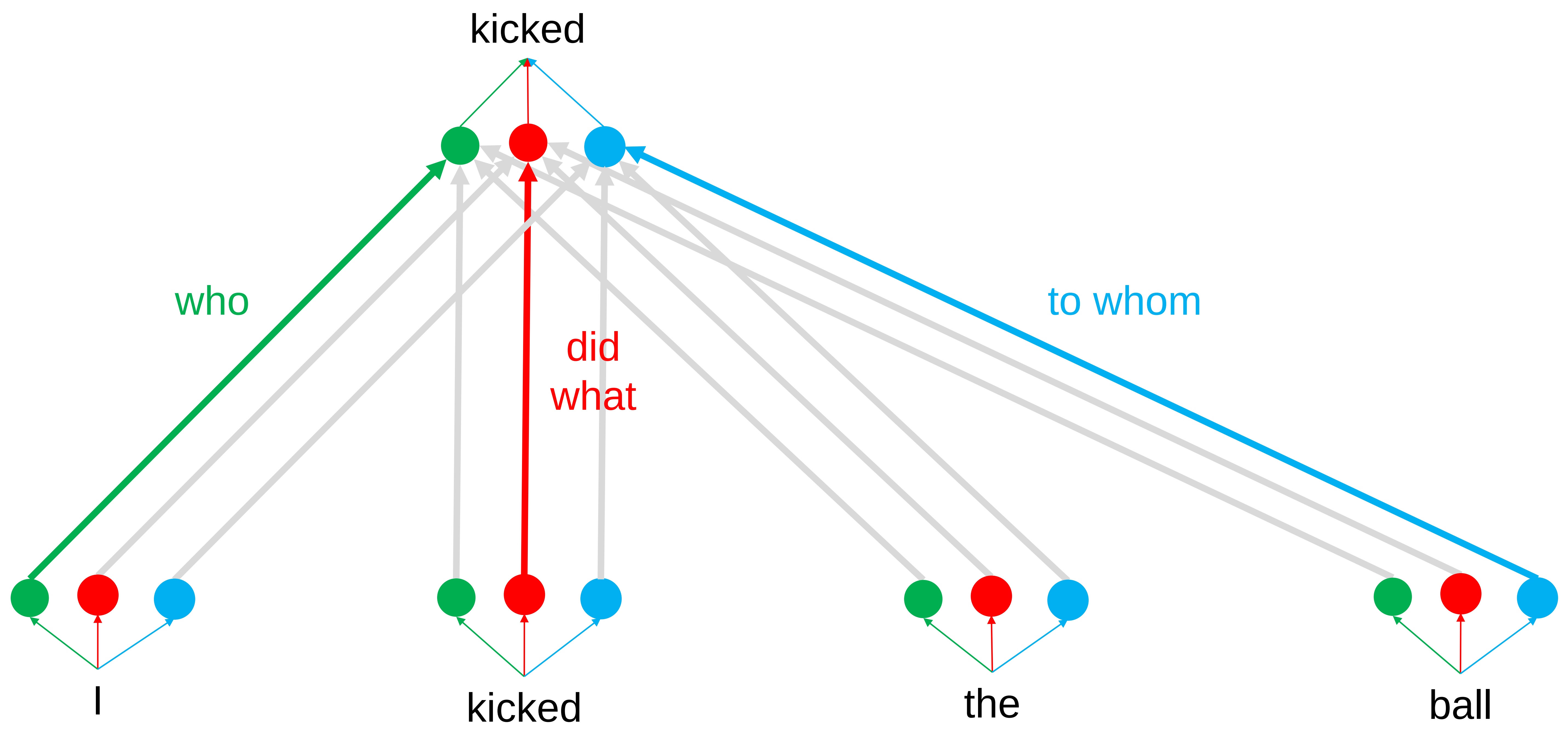
Attention Head: did what



Attention Head: to whom

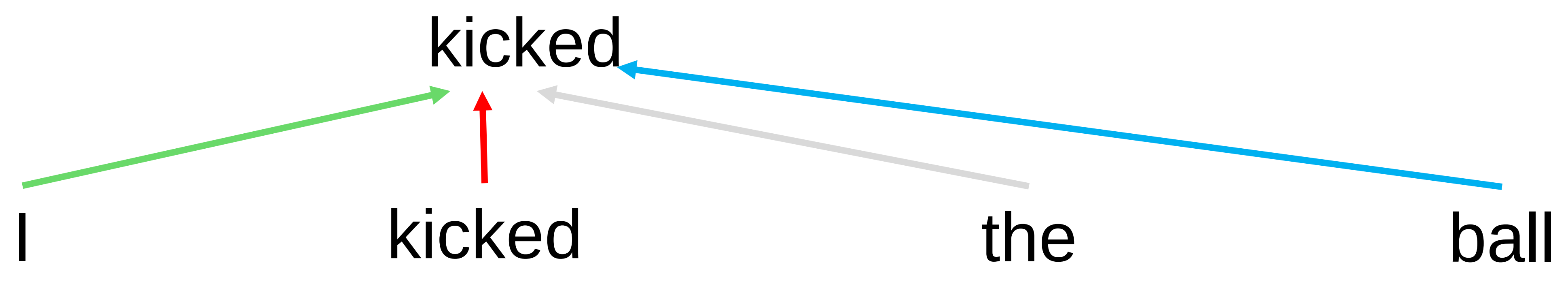


Multi-Head Attention



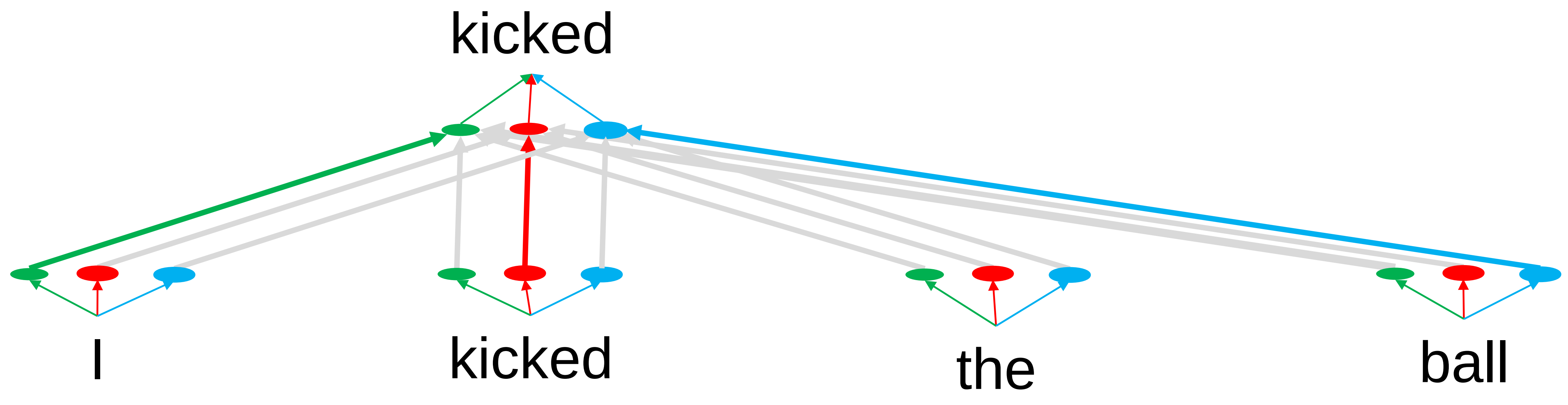
Comparison

- Convolution: different linear transformations by relative positions



- Attention: a weighted average

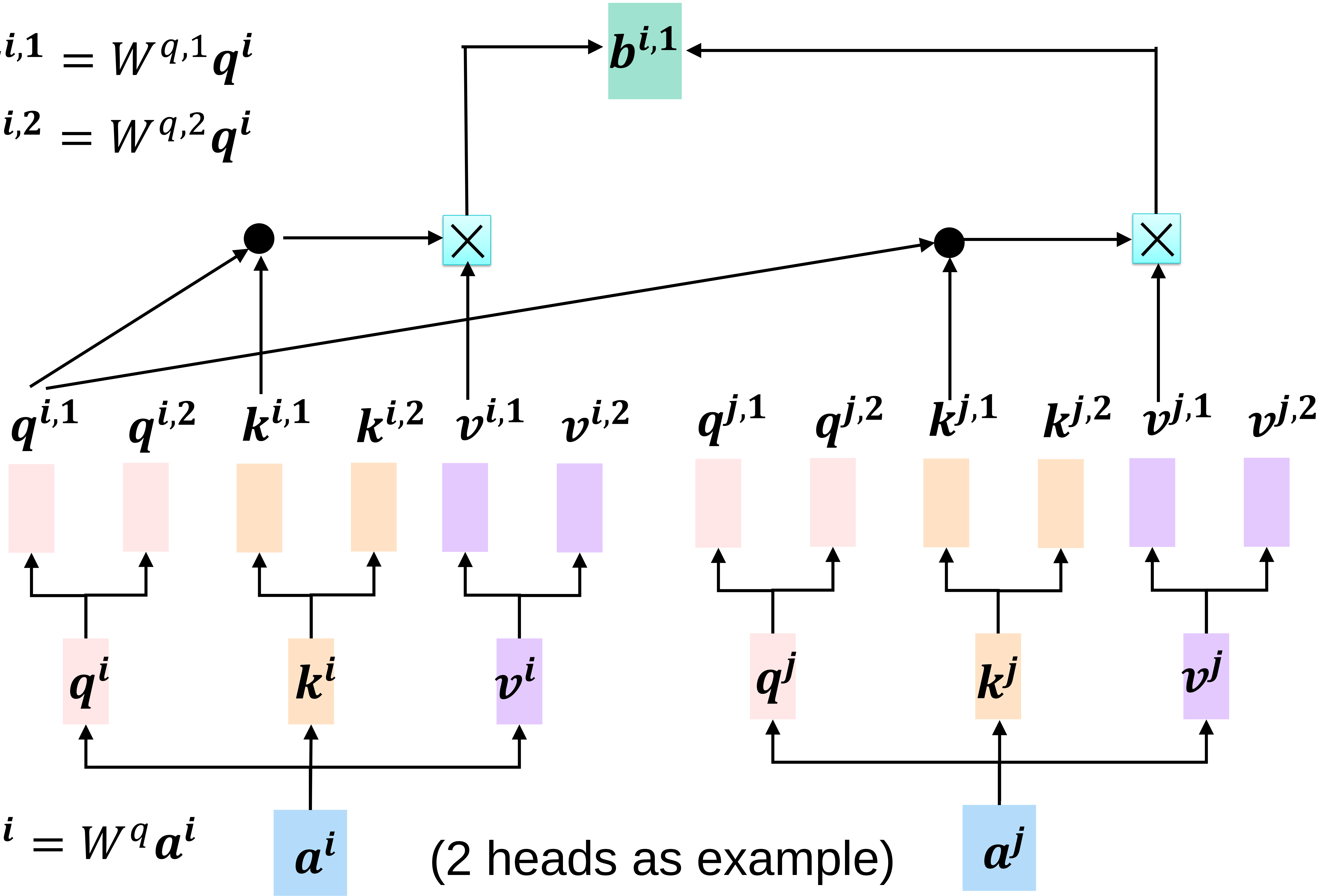
- Multi-Head Attention: parallel attention layers with different linear transformations on input/output



Multi-Head Attention

$$q^{i,1} = W^{q,1} q^i$$

$$q^{i,2} = W^{q,2} q^i$$

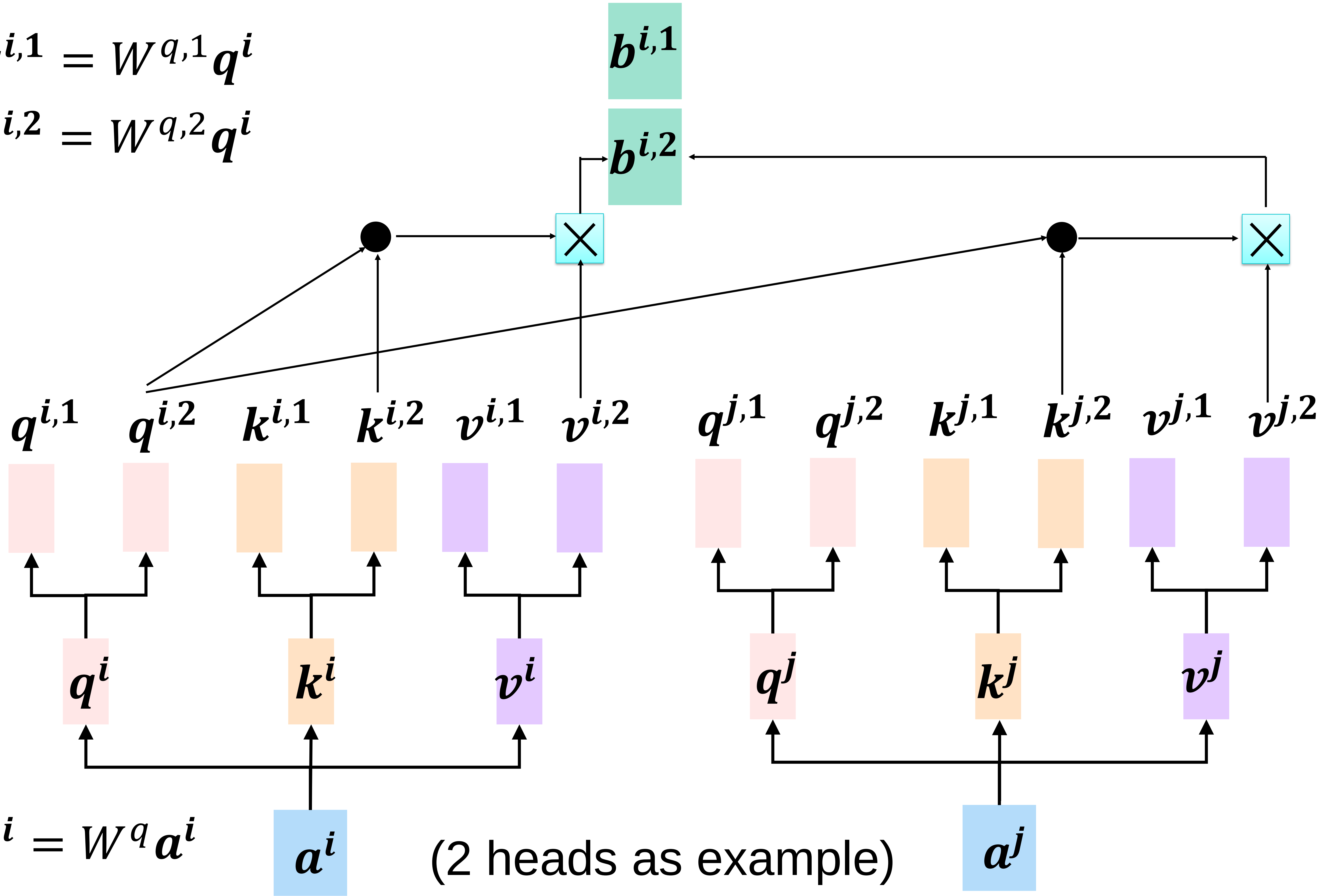


$$q^i = W^q a^i$$

Multi-Head Attention

$$q^{i,1} = W^{q,1} q^i$$

$$q^{i,2} = W^{q,2} q^i$$

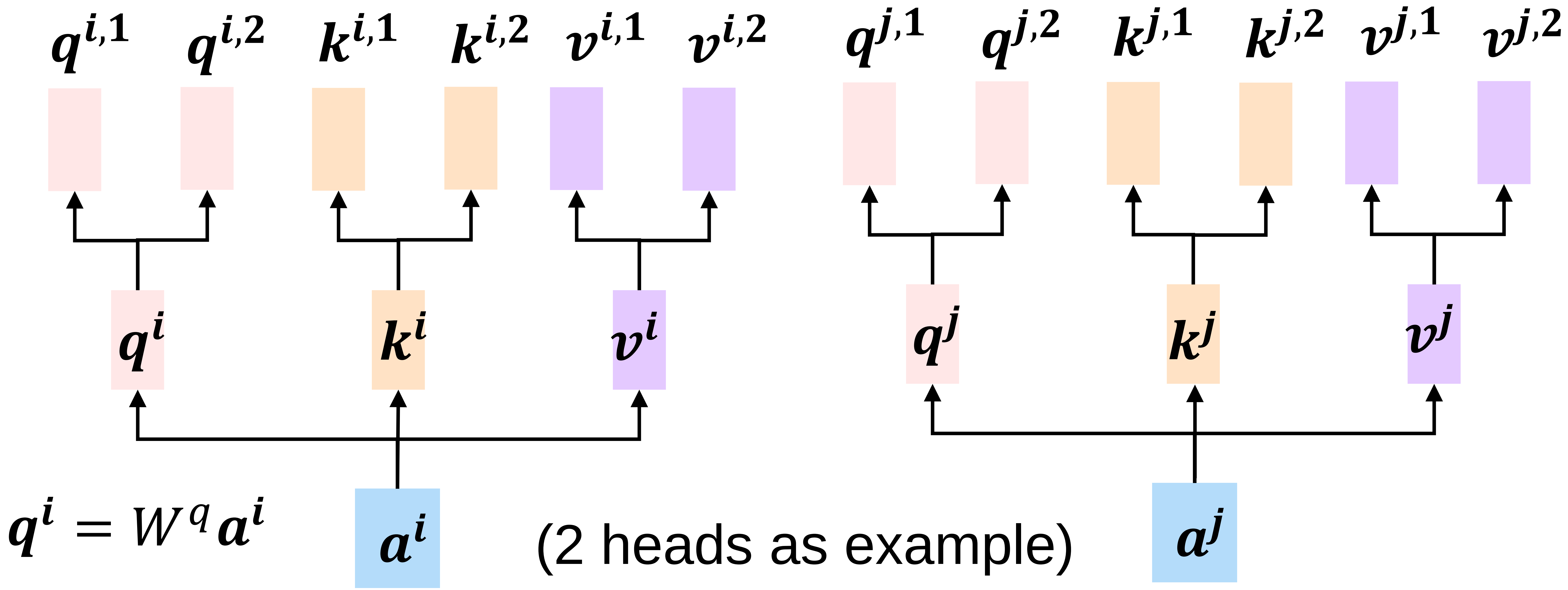


$$q^i = W^q a^i$$

(2 heads as example)

Multi-Head Attention

$$\boldsymbol{b}^i = W^O \begin{bmatrix} \boldsymbol{b}^{i,1} \\ \boldsymbol{b}^{i,2} \end{bmatrix}$$



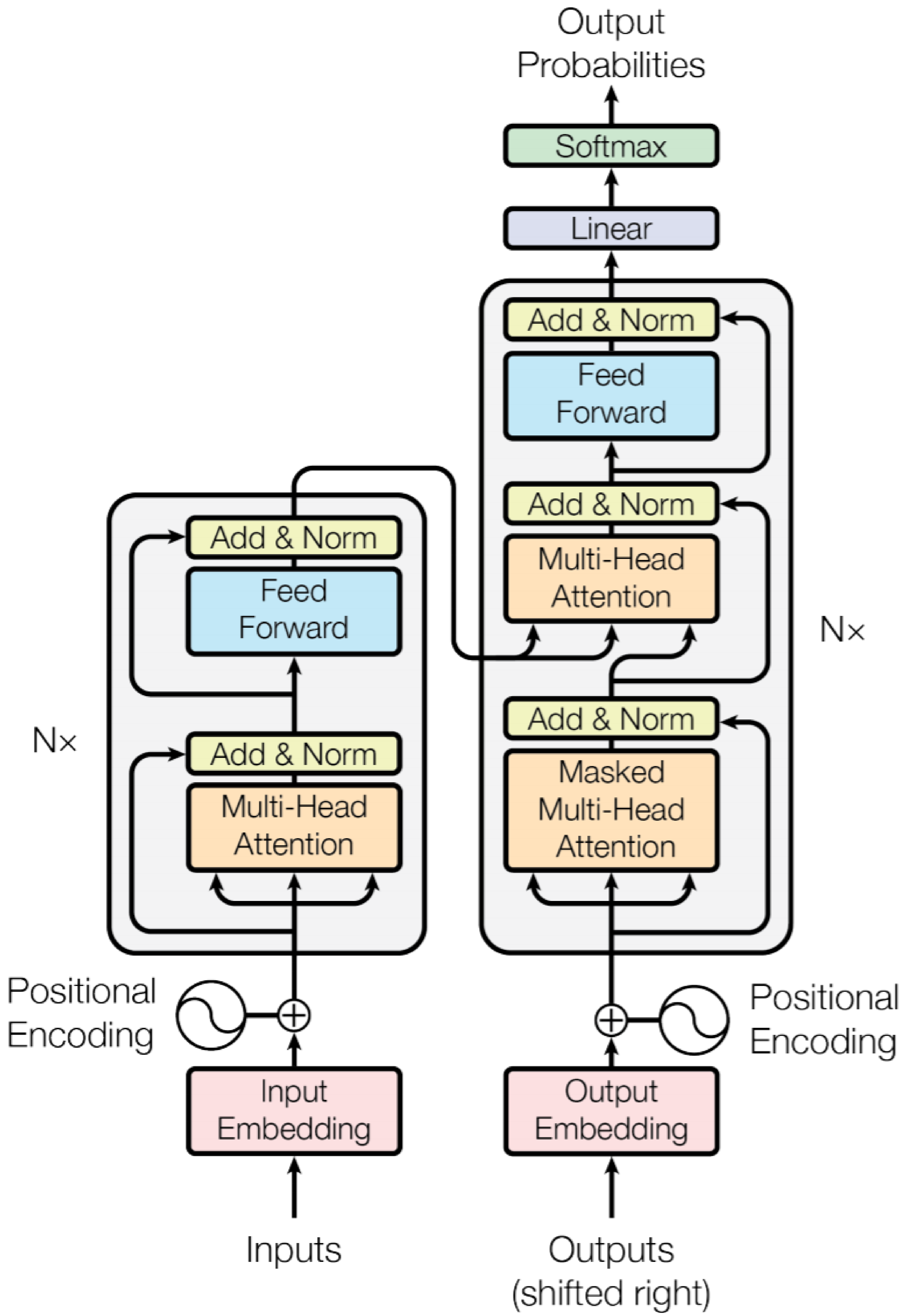
Sequence Encoding

Transformer

Transformer Overview

- Non-recurrent encoder-decoder for MT
- PyTorch explanation by Sasha Rush

http://nlp.seas.harvard.edu/2018/04/03/attention.html

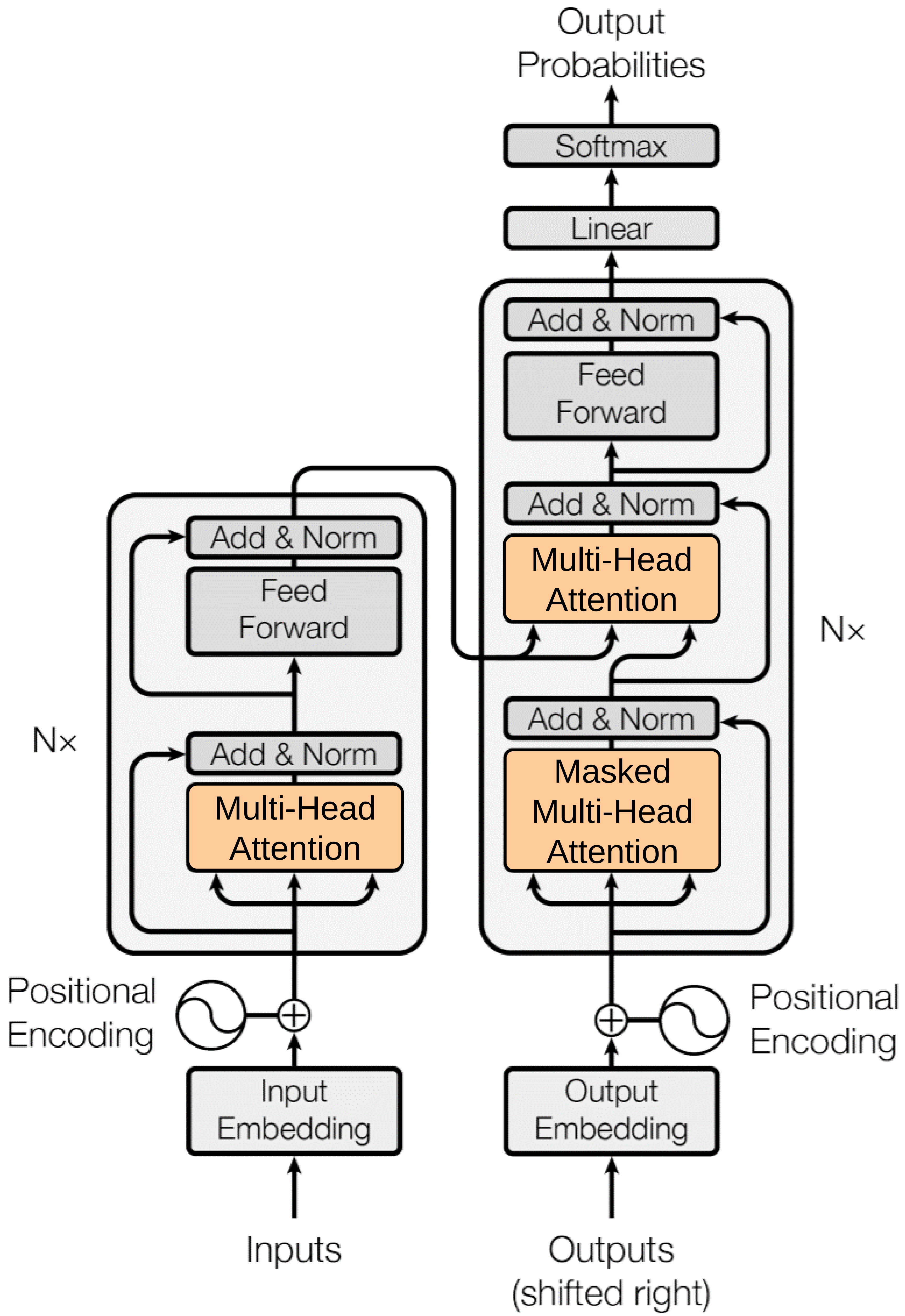


Vaswani et al., "Attention Is All You Need", in *NIPS*, 2017.

Transformer Overview

- Non-recurrent encoder-decoder for MT
- PyTorch explanation by Sasha Rush

  http://nlp.seas.harvard.edu/2018/04/03/attention.html



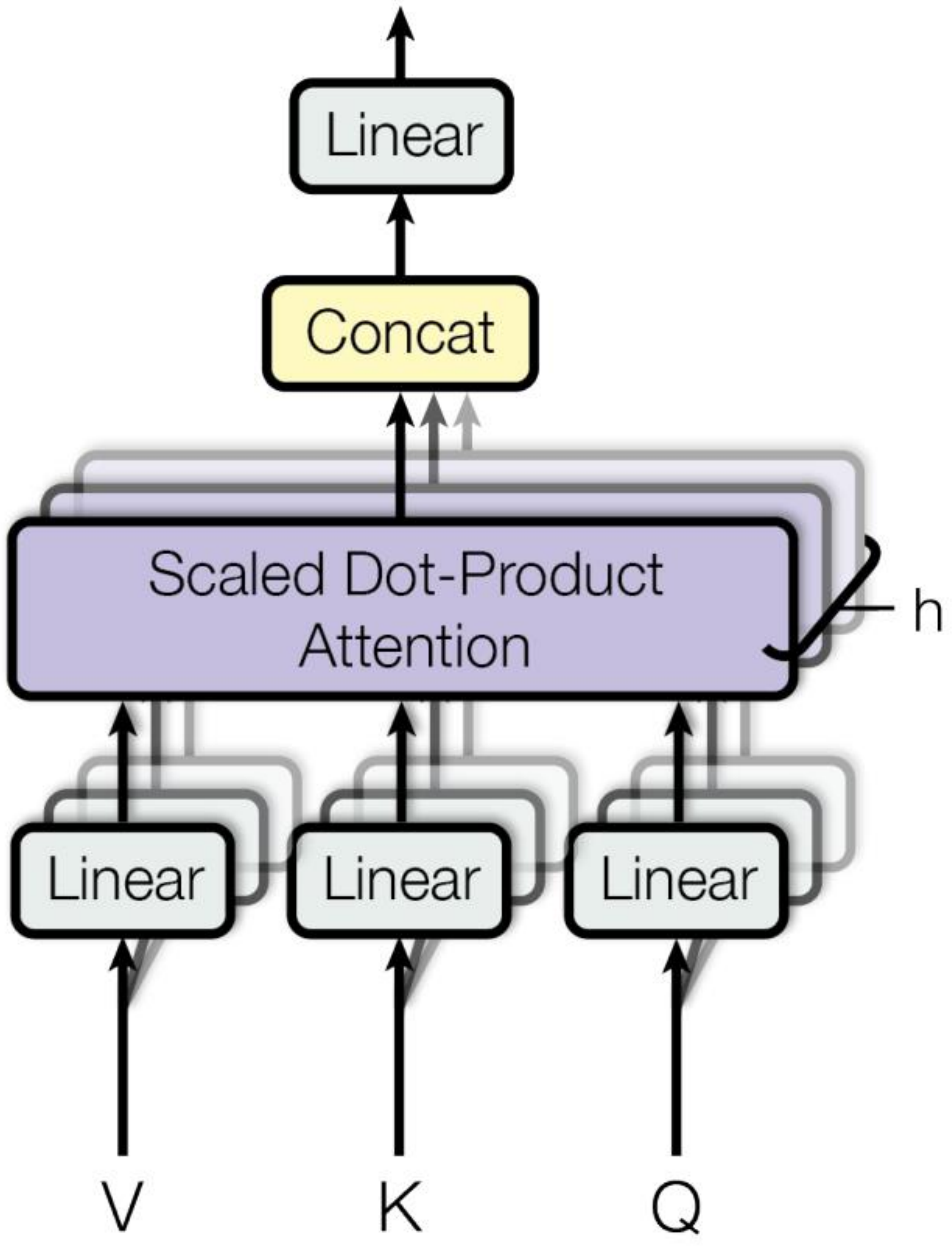
Vaswani et al., “Attention Is All You Need”, in *NIPS*, 2017.

Multi-Head Attention

- Idea: allow words to interact with one another
- Model
  - Map V, K, Q to lower dimensional spaces
  - Apply attention, concatenate outputs
  - Linear transformation

$$\begin{aligned}\mathrm{MultiHead}(Q,K,V)\\ &= \mathrm{Concat}(\mathrm{head}_1,\cdots,\mathrm{head}_h)W^O\end{aligned}$$

$$\mathrm{head}_i = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



Vaswani et al., "Attention Is All You Need", in *NIPS*, 2017.

Scaled Dot-Product Attention

- Problem: when d_k gets large, the variance of $q^T k$ increases

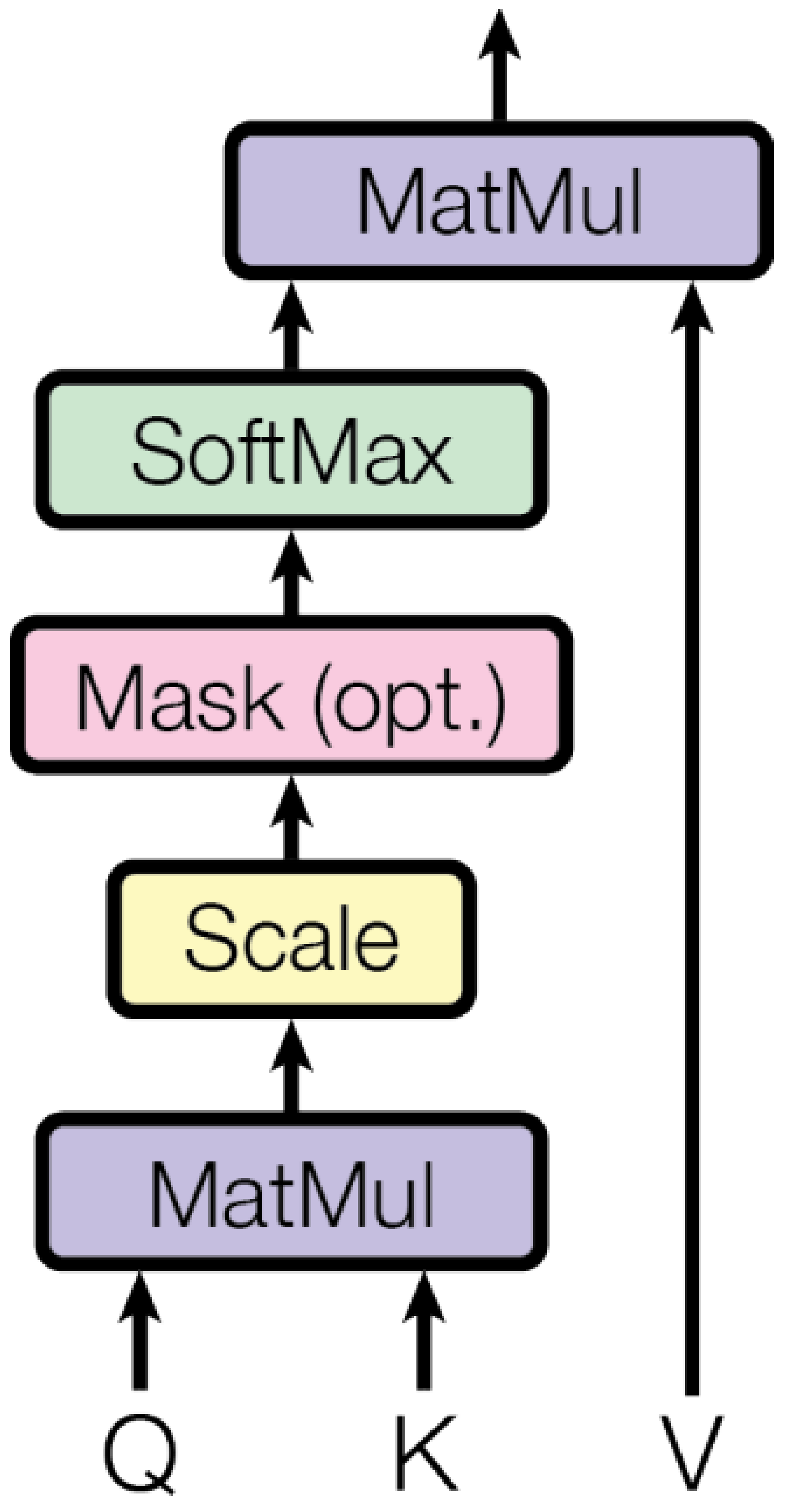
→ q and k are random variables with mean 0 and variance 1

→ $q^T k$ has mean 0 and variance d_k

→ variance 1 is preferred

- Solution: scale by $\sqrt{d_k}$

$$A(Q, K, V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

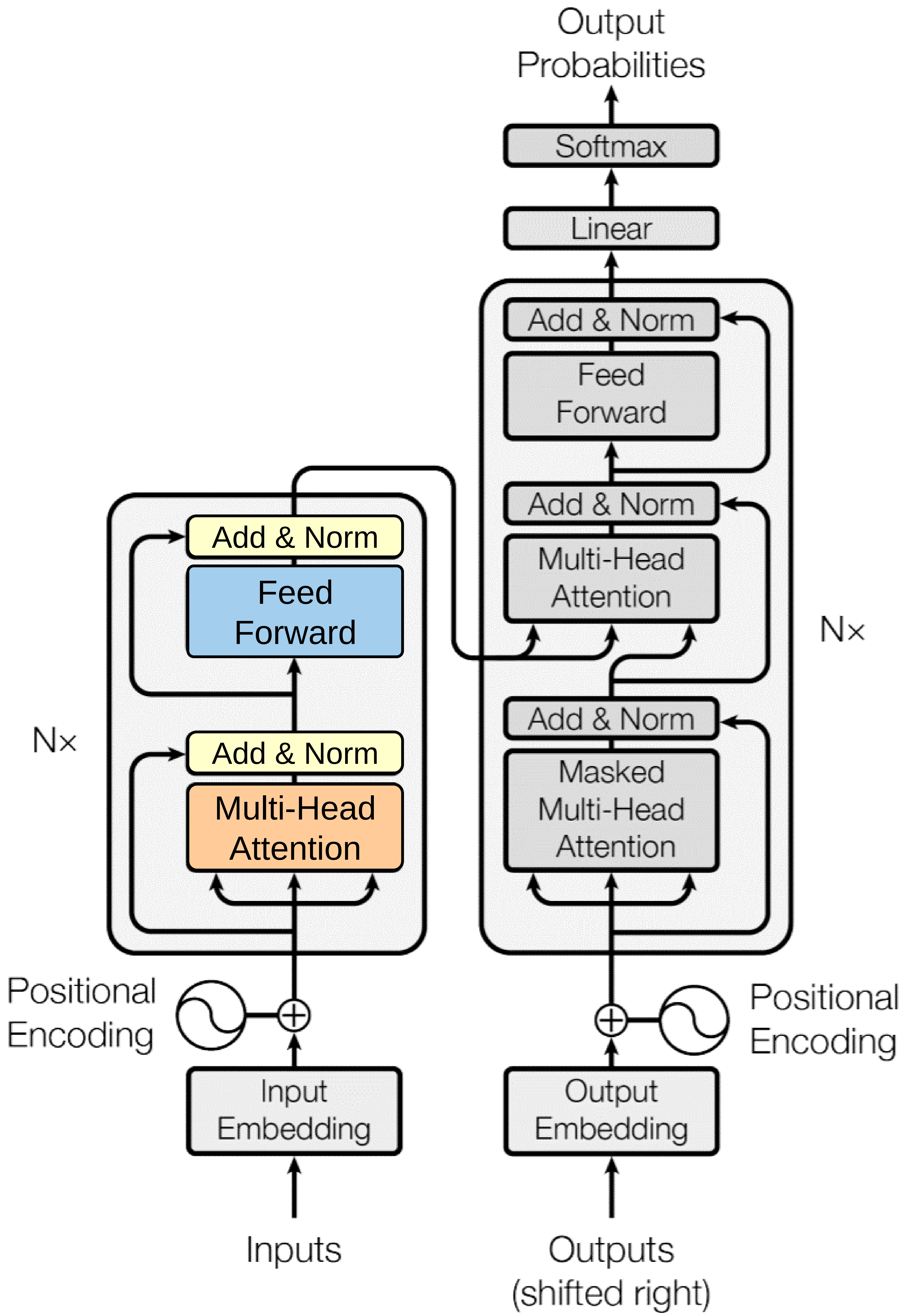


Vaswani et al., "Attention Is All You Need", in *NIPS*, 2017.

Transformer Overview

- Non-recurrent encoder-decoder for MT
- PyTorch explanation by Sasha Rush

http://nlp.seas.harvard.edu/2018/04/03/attention.html



Vaswani et al., "Attention Is All You Need", in *NIPS*, 2017.

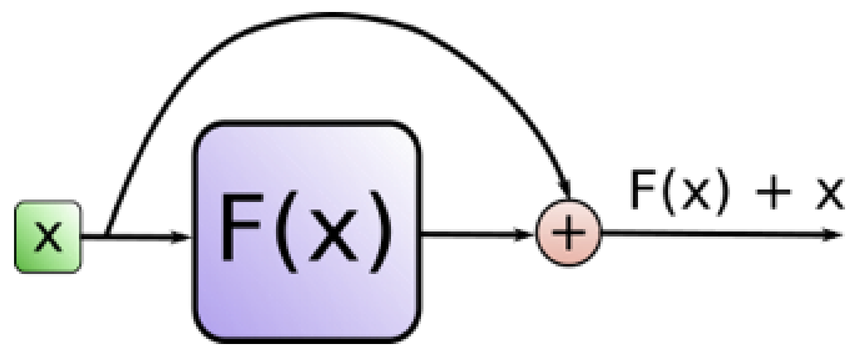
Transformer Encoder Block

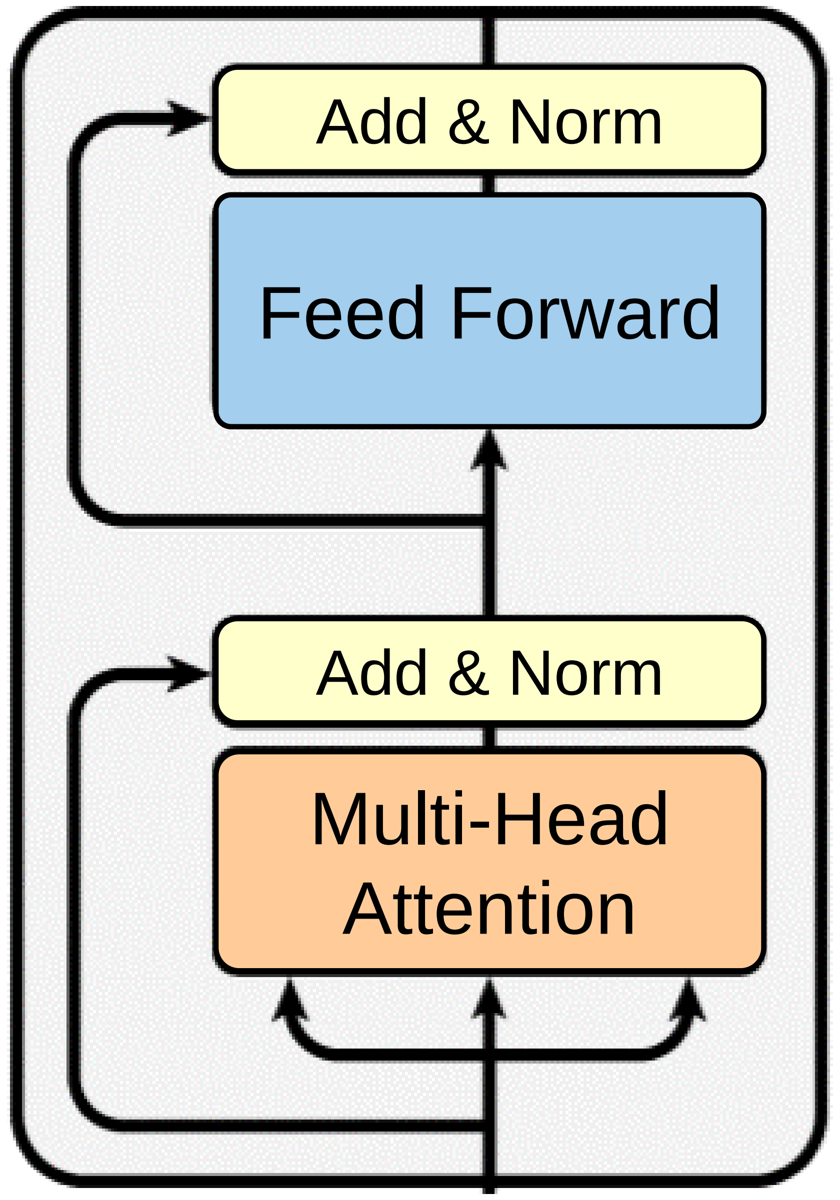
- Each block has
  - multi-head attention
  - 2-layer feed-forward NN (w/ ReLU)
- Both parts contain
  - Residual connection
  - Layer normalization (LayerNorm)

Change input to have 0 mean and 1 variance per layer & per training point

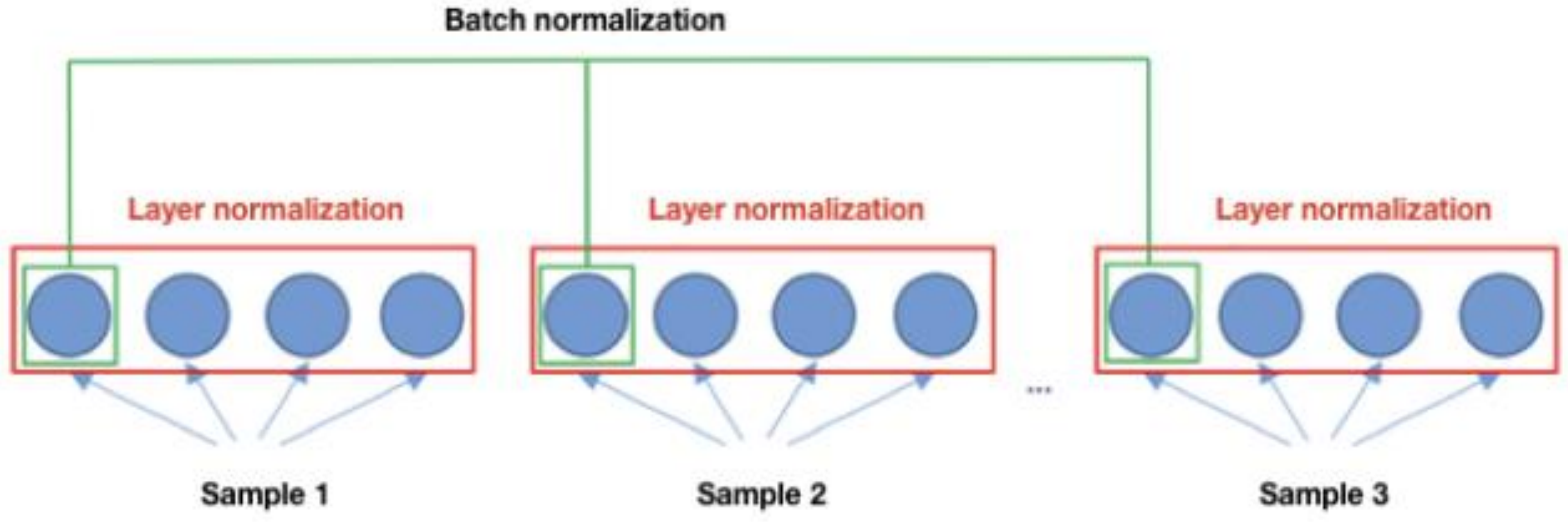
→ LayerNorm(x + sublayer(x))

$$H(x) = g(x) = x + F(x)$$





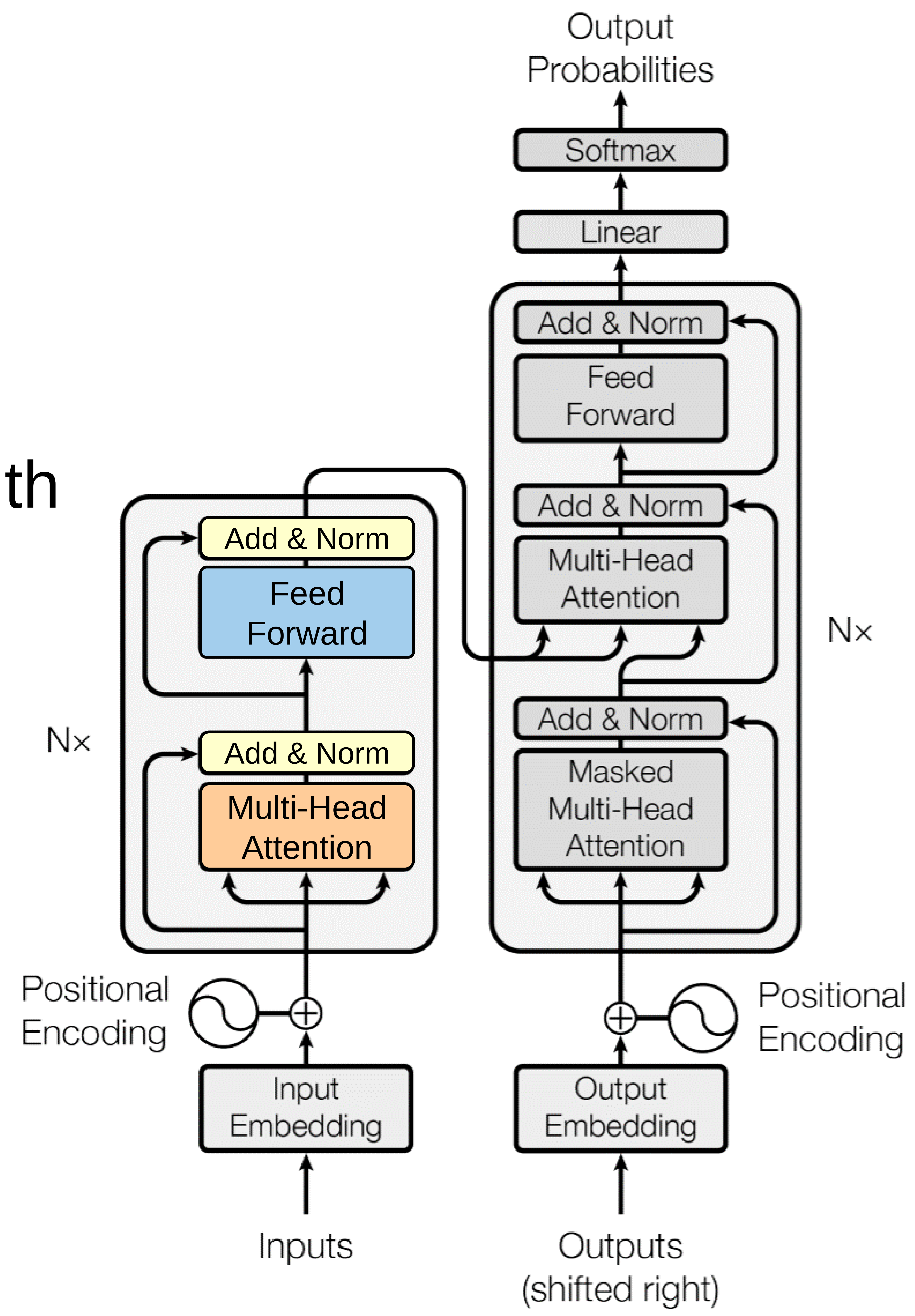
$$\mu^l = \frac{1}{H}\sum_{i=1}^{H} a_i^l \quad \sigma^l = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(a_i^l - \mu^l)^2} \quad h_i = f\left(\frac{g_i}{\sigma_i}(a_i - \mu_i) + b_i\right)$$



https://medium.com/@bgg/seq2seq-pay-attention-to-self-attention-part-2-中文版-ef2ddf8597a4

Encoder Input

- Problem: temporal information is missing
- Solution: **positional encoding** allows words at different locations to have different embeddings with fixed dimensions



https://medium.com/@bgg/seq2seq-pay-attention-to-self-attention-part-2-中文版-ef2ddf8597a4

Positional Encoding

- Criteria for positional encoding
  - Unique encoding for each position
  - Deterministic
  - Distance between neighboring positions should be the same
  - Model can easily generalize to longer sentences

Positional Encoding

- Criteria for positional encoding
  - ✓ Unique encoding for each position
  - ✓ Deterministic
  - ✓ Distance between neighboring positions should be the same
  - Model can easily generalize to longer sentences
- Idea 1: $\mathrm{PE}(pos) = pos$
  - A value to indicate the word’s position
  - Larger value (longer sentence) may not be easily generalized ☹

Positional Encoding

- Criteria for positional embeddings
  - ✓ Unique encoding for each position
  - ✓ Deterministic
  - ✓ Distance between neighboring positions should be the same
  - Model can easily generalize to longer sentences
- Idea 2: 1-hot encoding
  - A d-dim vector to encode d positions
  - Cannot generalize to longer sentences ☹

$\underbrace{[1, 0, 0, \ldots, 0]}_{\text{d-dim}}$ ➡ only represent the sequences with the length <= d

Positional Encoding

- Criteria for positional encoding
  - ✓ Unique encoding for each position
  - ✓ Deterministic
  - Distance between neighboring positions should be the same
  - ✓ Model can easily generalize to longer sentences
- Idea 3: $\mathrm{PE}(pos) = \frac{pos}{\mathrm{len}}$
  - The normalized value of the position (0~1)
  - Distances may differ in sentences with different lengths ☹

	w_1	w_2	w_3	w_4
PE	0.25	0.50	0.75	1.00

$\xleftrightarrow{0.25}$ (between 0.25 and 0.50): 0.25

	w_1	w_2	w_3	w_4	$\cdots$	w_{10}
PE	0.10	0.20	0.30	0.40		1.00

0.10

Sinusoidal Positional Encoding

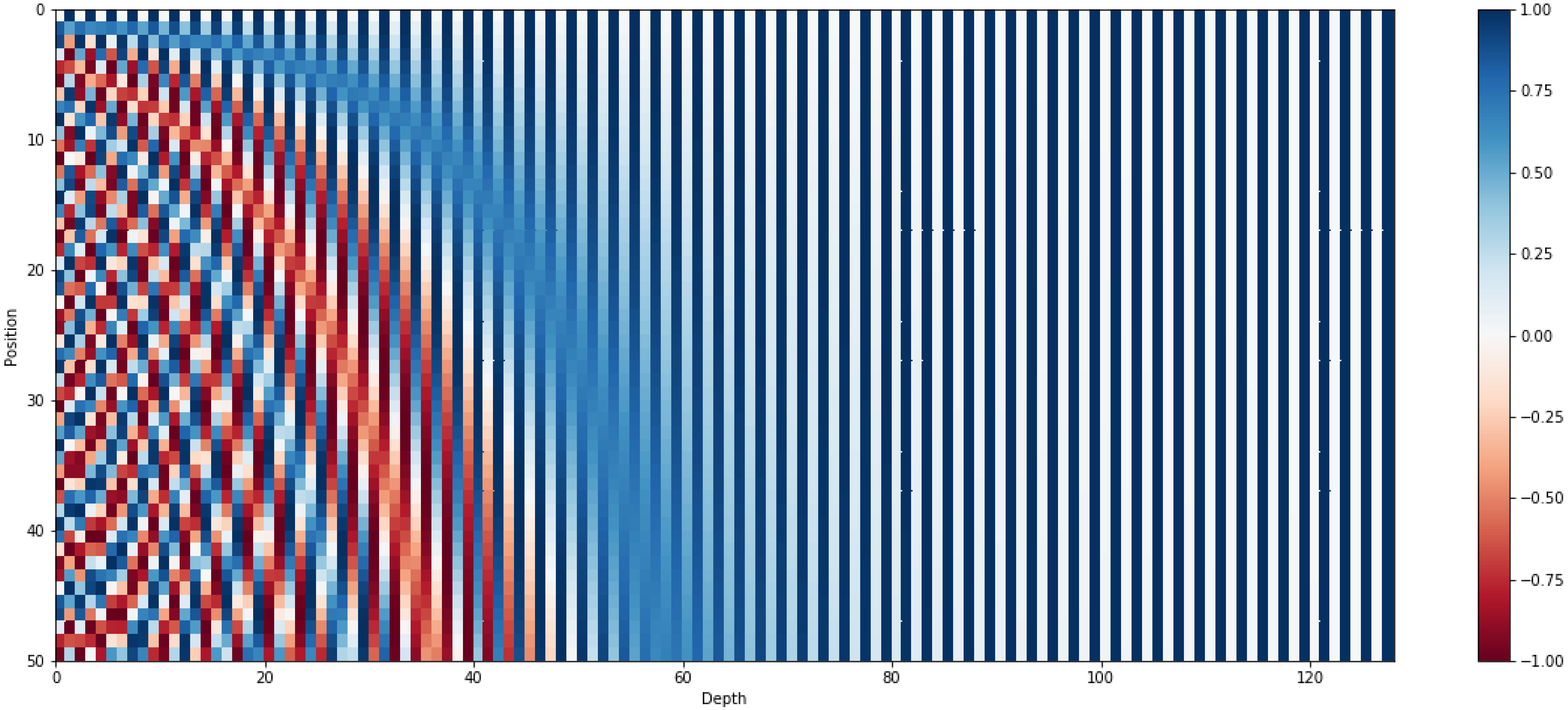
- Criteria for positional embeddings
  - ✓ Unique encoding for each position
  - ✓ Deterministic
  - ✓ Distance between neighboring positions should be the same
  - ✓ Model can easily generalize to longer sentences
- Idea:

$$\mathrm{PE}(pos, 2i) = sin(\frac{pos}{100000^{2i/d}})$$

$$\mathrm{PE}(pos, 2i+1) = cos(\frac{pos}{100000^{2i/d}})$$

  - A d-dim vector to represent positions

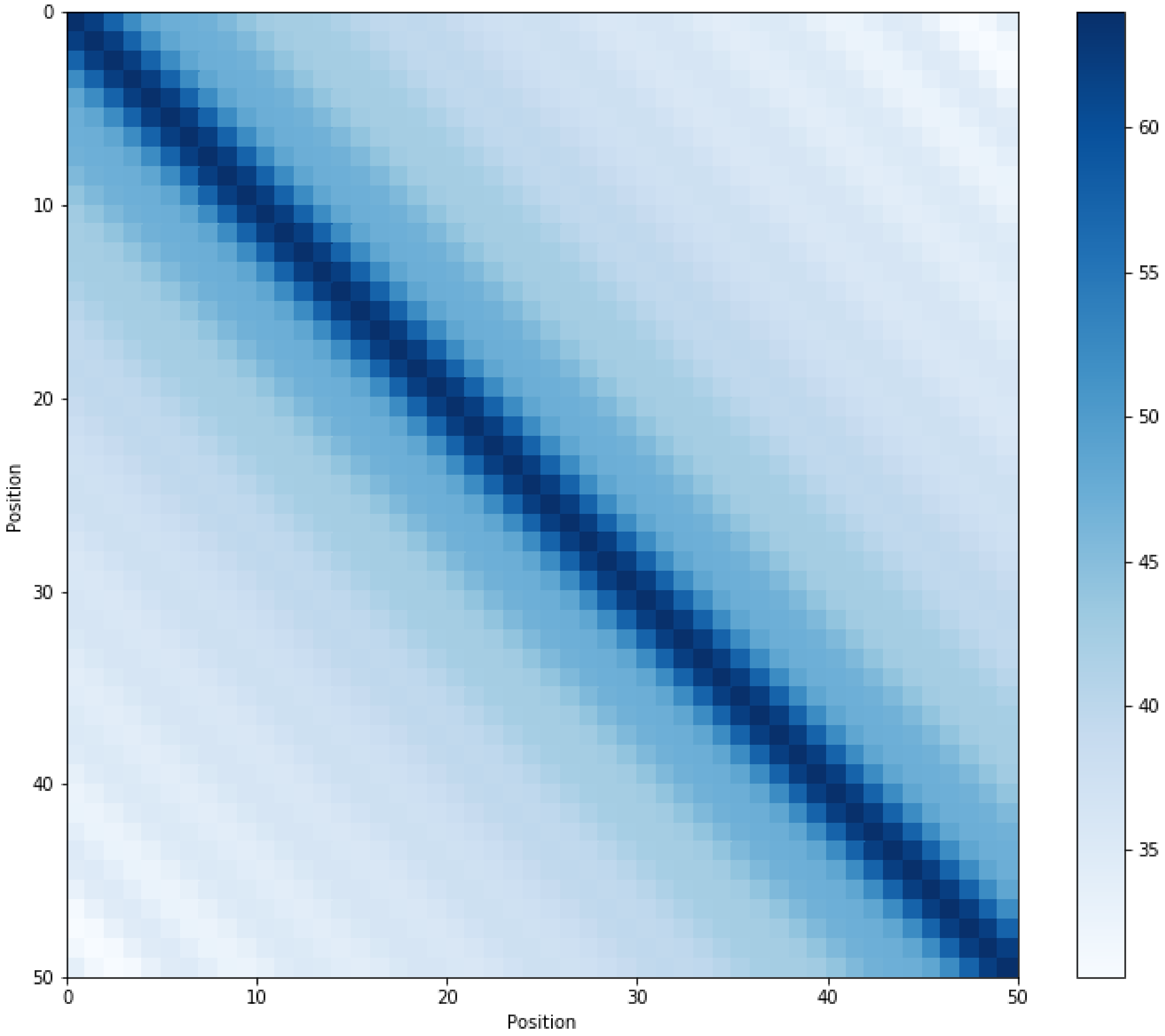
Sinusoidal Positional Encoding



Transformer Architecture: The Positional Encoding - Amirhossein Kazemnejad's Blog

Sinusoidal Positional Encoding

- Distance between neighboring positions
  - symmetrical
  - decay nicely with time

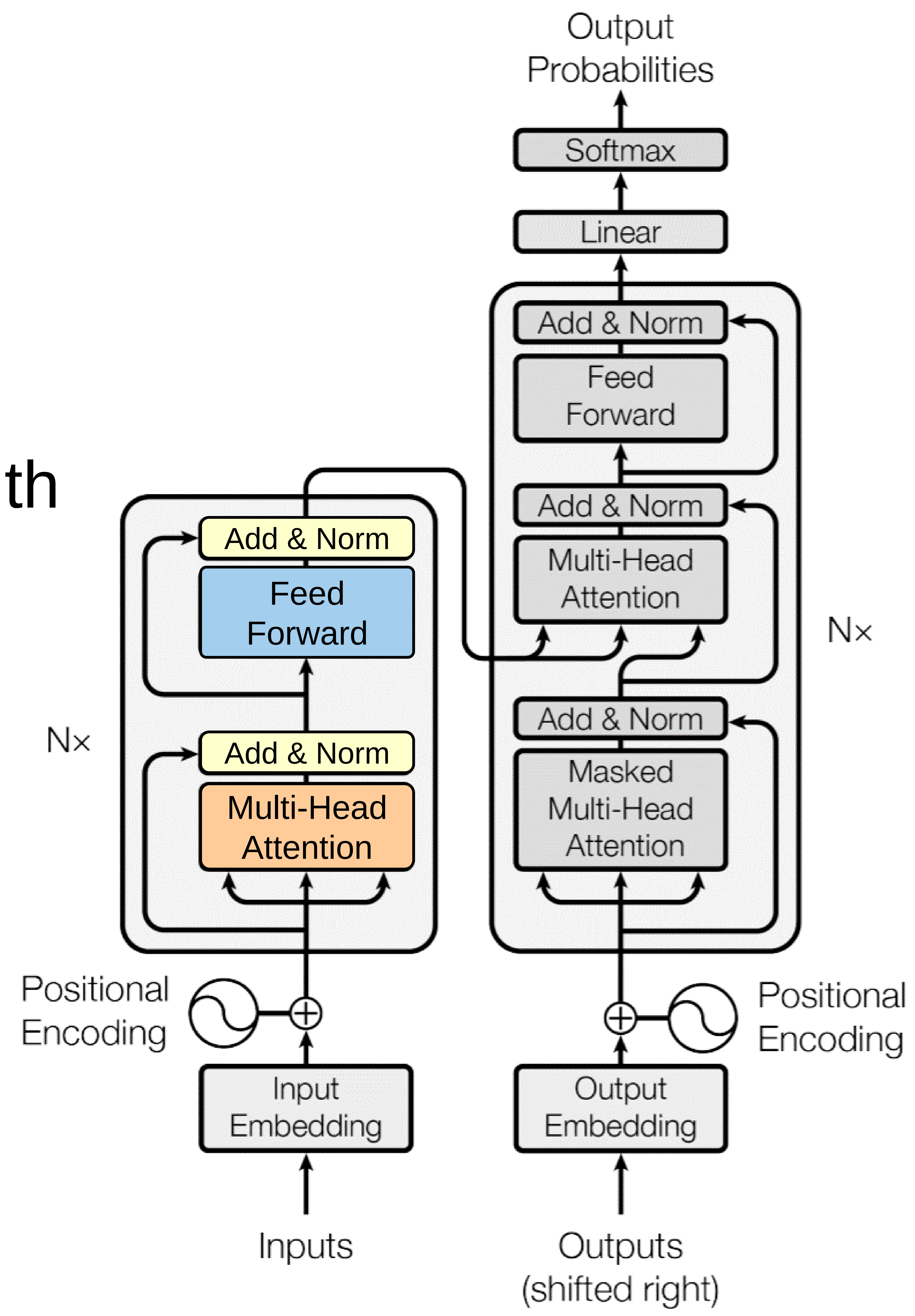


Dot product of position embeddings for all time-steps

[Transformer Architecture: The Positional Encoding - Amirhossein Kazemnejad's Blog]

Encoder Input

- Problem: temporal information is missing
- Solution: **positional encoding** allows words at different locations to have different embeddings with fixed dimensions



https://medium.com/@bgg/seq2seq-pay-attention-to-self-attention-part-2-中文版-ef2ddf8597a4

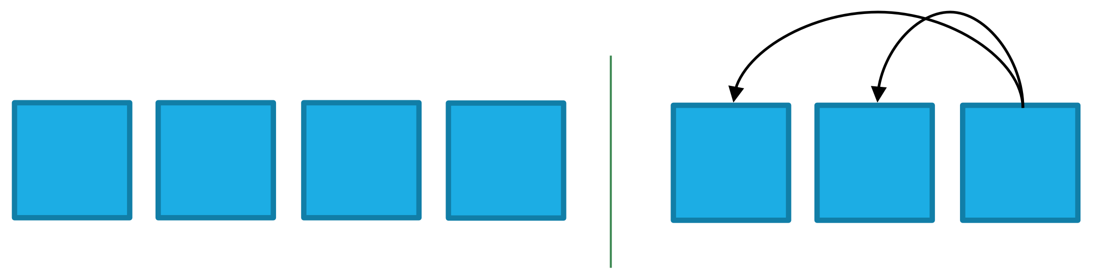
Multi-Head Attention Details

encoder self attention

1. **Multi-head Attention**
2. **Query=Key=Value**

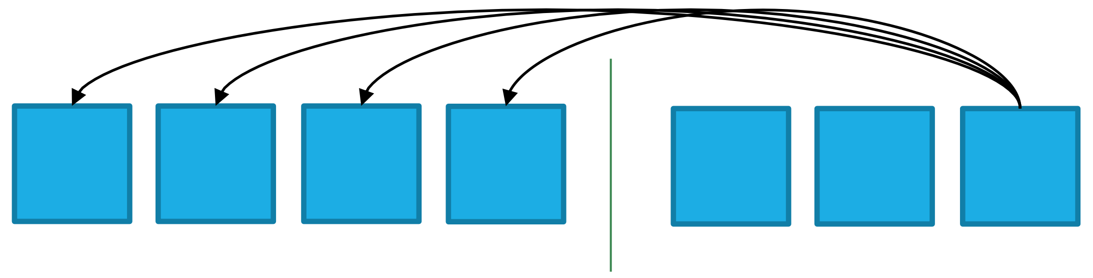
decoder self attention

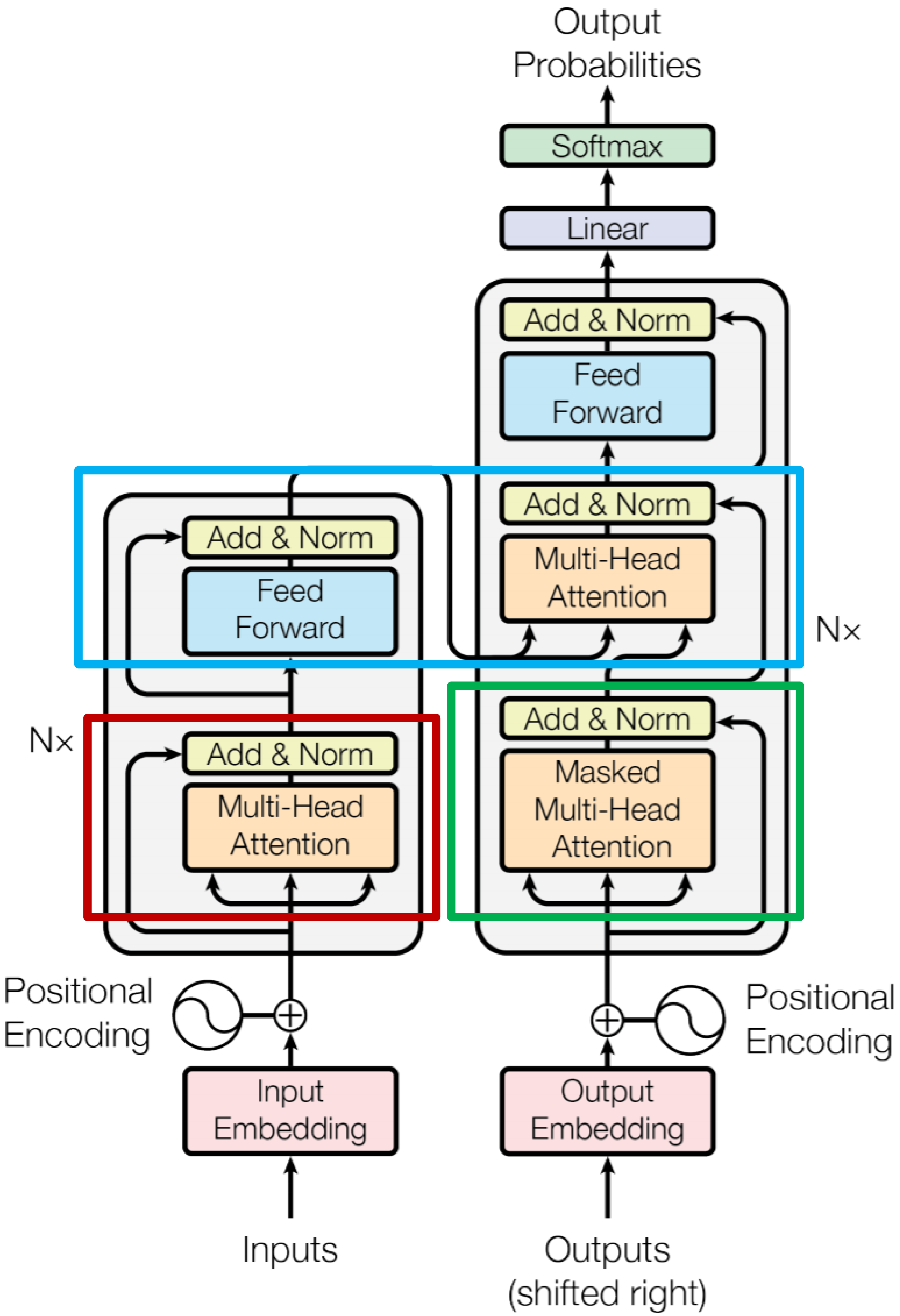
1. **Masked Multi-head Attention**
2. **Query=Key=Value**



encoder-decoder attention

1. **Multi-head Attention**
2. **Encoder Self attention=Key=Value**
3. **Decoder Self attention=Query**





https://medium.com/@bgg/seq2seq-pay-attention-to-self-attention-part-2-中文版-ef2ddf8597a4

Training Tips

- Byte-pair encodings (BPE)
- Checkpoint averaging
- ADAM optimizer with learning rate changes
- Dropout during training at every layer just before adding residual
- Label smoothing
- Auto-regressive decoding with beam search and length penalties

MT Experiments

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Vaswani et al., "Attention Is All You Need", in *NIPS*, 2017.

Parsing Experiments

Parser	Training	WSJ 23 F1
Vinyals & Kaiser el al. (2014) [37]	WSJ only, discriminative	88.3
Petrov et al. (2006) [29]	WSJ only, discriminative	90.4
Zhu et al. (2013) [40]	WSJ only, discriminative	90.4
Dyer et al. (2016) [8]	WSJ only, discriminative	91.7
Transformer (4 layers)	WSJ only, discriminative	91.3
Zhu et al. (2013) [40]	semi-supervised	91.3
Huang & Harper (2009) [14]	semi-supervised	91.3
McClosky et al. (2006) [26]	semi-supervised	92.1
Vinyals & Kaiser el al. (2014) [37]	semi-supervised	92.1
Transformer (4 layers)	semi-supervised	92.7
Luong et al. (2015) [23]	multi-task	93.0
Dyer et al. (2016) [8]	generative	93.3

Vaswani et al., "Attention Is All You Need", in *NIPS*, 2017.

Concluding Remarks

- **Non-recurrence** model is easy to parallelize
- **Multi-head attention** captures different aspects by interacting between words
- **Positional encoding** captures location information
- Each transformer block can be applied to diverse tasks