Transformer

李宏毅

Hung-yi Lee

Transformer

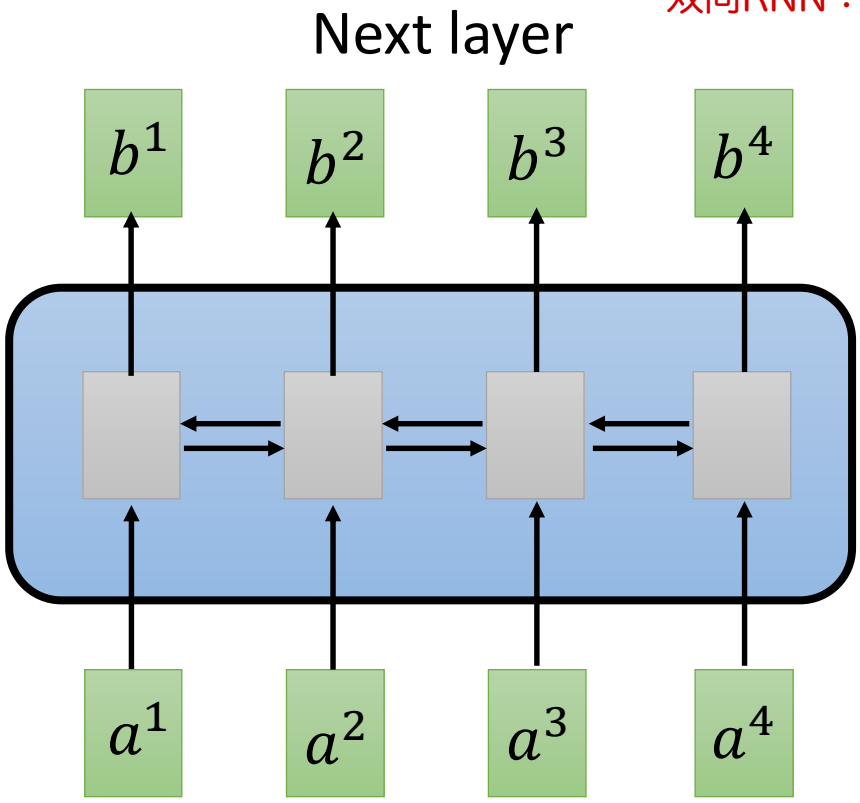
Seq2seq model with “Self-attention”

问题：RNN 不容易被并行化.

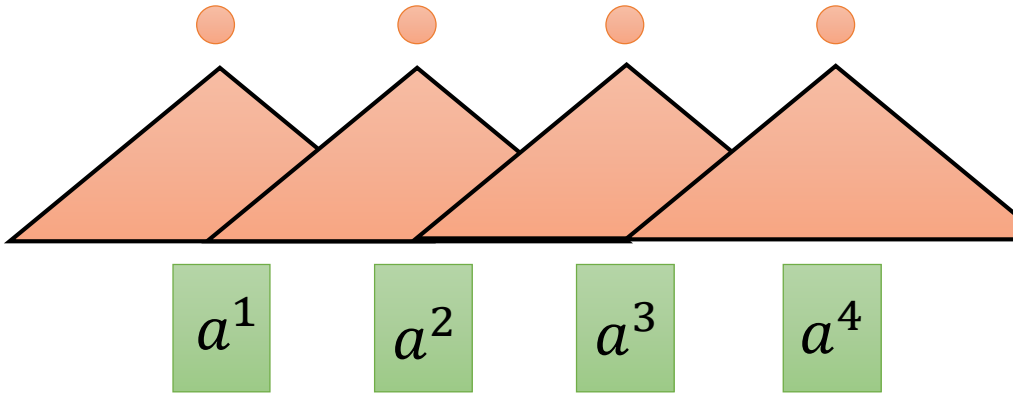
Sequence

单向RNN：输出 b3 时，看过 b1 b2；输出 b4时，看过 b1 b2 b3 。

双向RNN：输出 b1/b2/b3/b4时，a1 a2 a3 a4 全部都看过了。



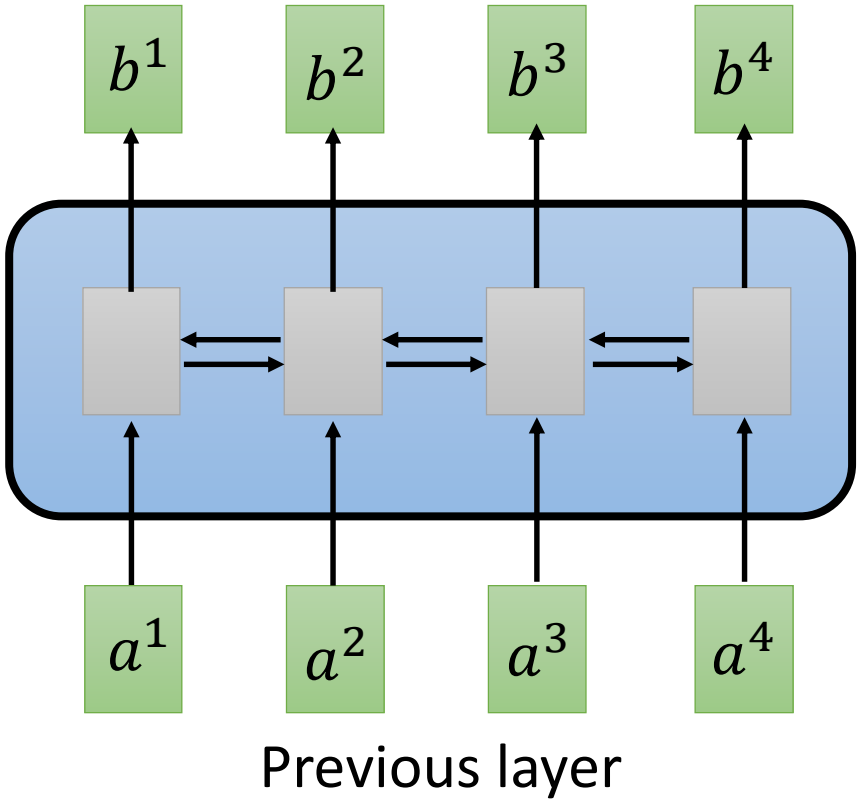
Hard to parallel !



Using CNN to replace RNN

使用 CNN 取代 RNN

Sequence



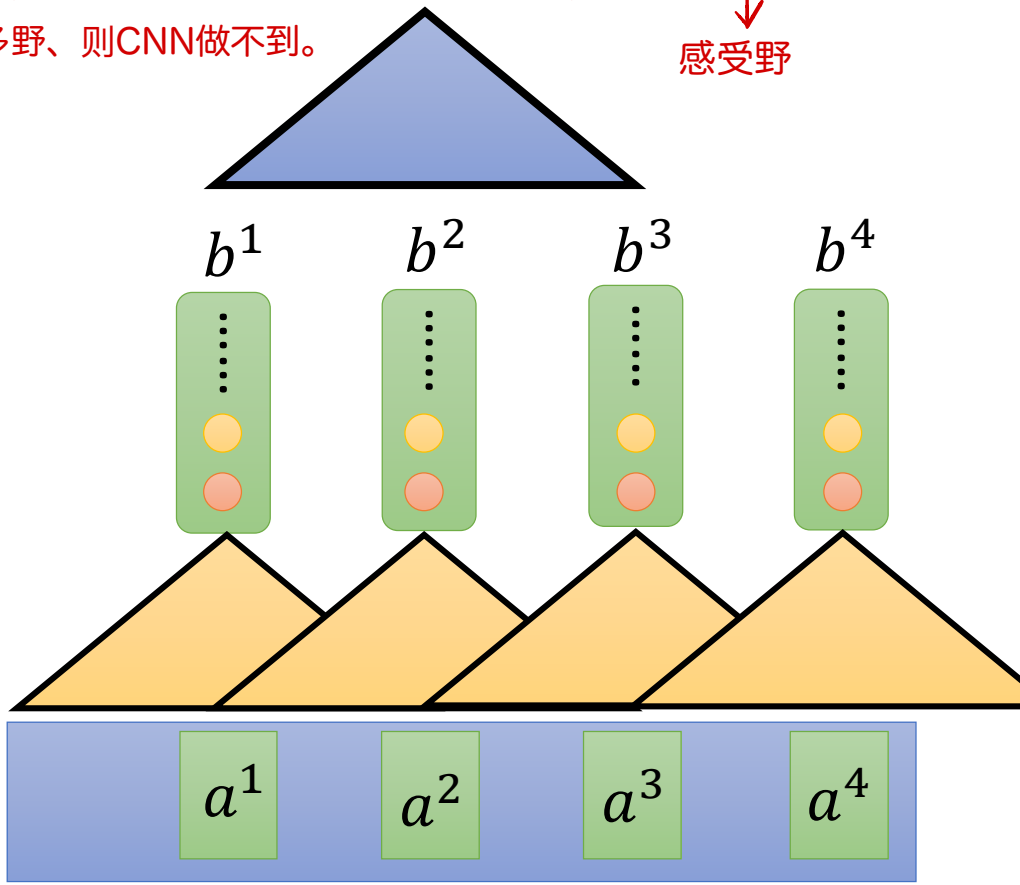
Hard to parallel

随着层数越来越多，感受野越来越大

Filters in higher layer can consider longer sequence

缺点：随着层数的增加，感受野才大。但起始层感受野很小，如果你开始就想要大的感多野、则CNN做不到。

感受野



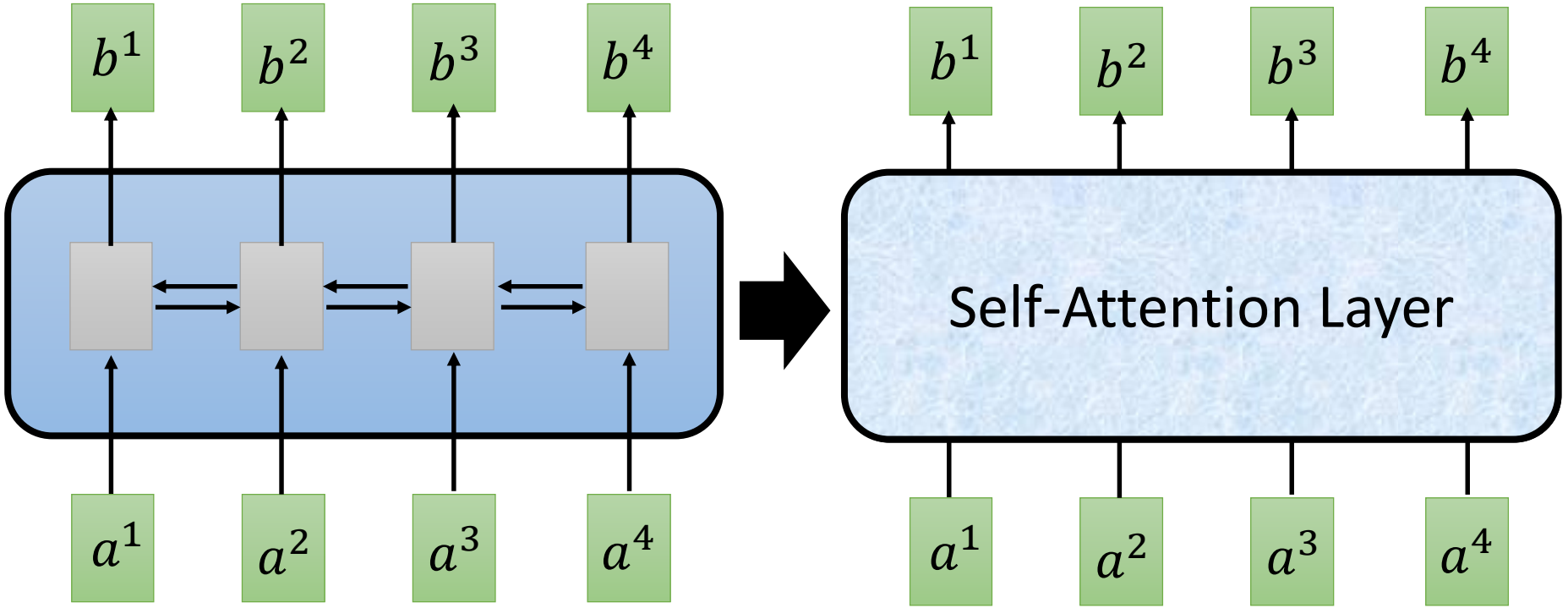
Using CNN to replace RNN (CNN can parallel)

Self-Attention

b^i is obtained based on the whole input sequence.

b^1, b^2, b^3, b^4 can be parallelly computed.

1. 输入 sequence，输出 sequence。
2. 每个输出，都看过了输入的所有内容，而不仅仅是 CNN中的部分内容。



You can try to replace any thing that has been done by RNN with self-attention.

Self-attention

https://arxiv.org/abs/1706.03762

q: query (to match others)

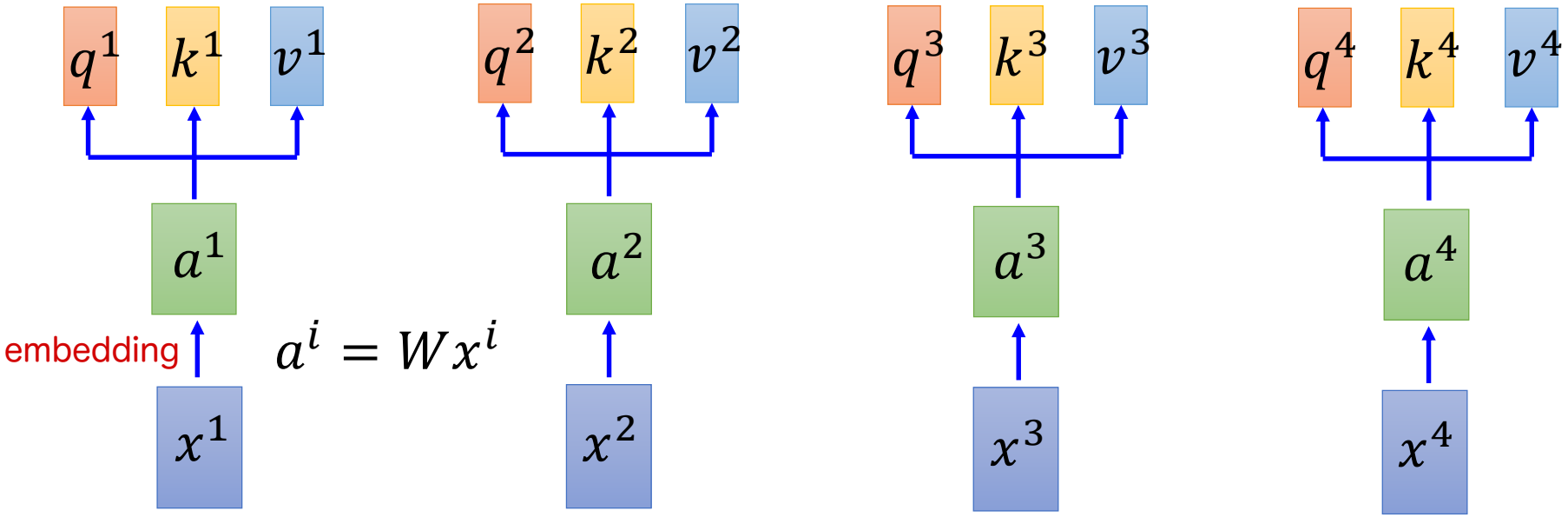
$$q^i = W^q a^i$$

k: key (to be matched)

$$k^i = W^k a^i$$

v: information to be extracted

$$v^i = W^v a^i$$



Self-attention

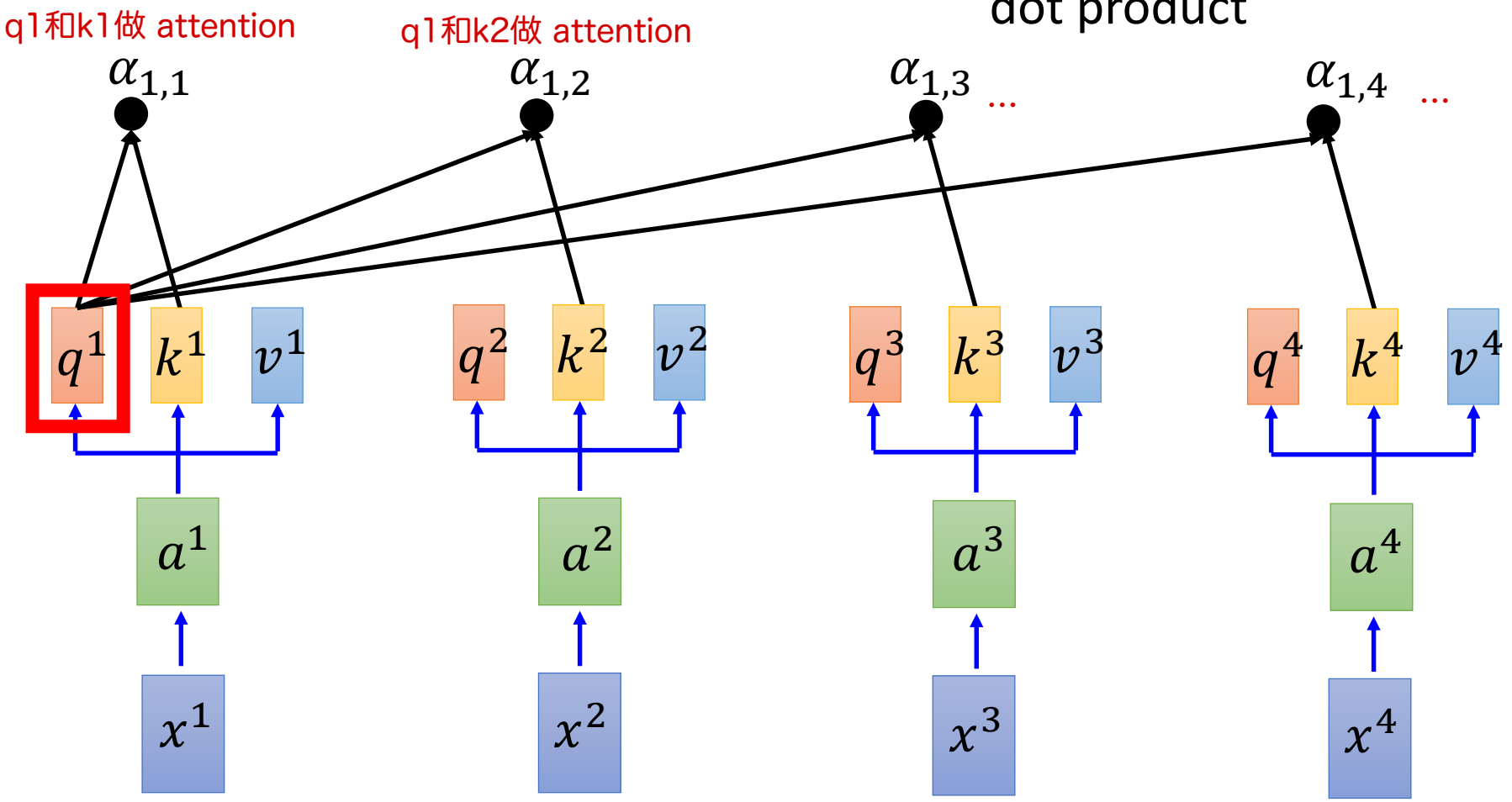
随着 dim 越大， variance 越大。所以除以根号 d。

d is the dim of q and k

拿每個 query q 去對每個 key k 做 attention

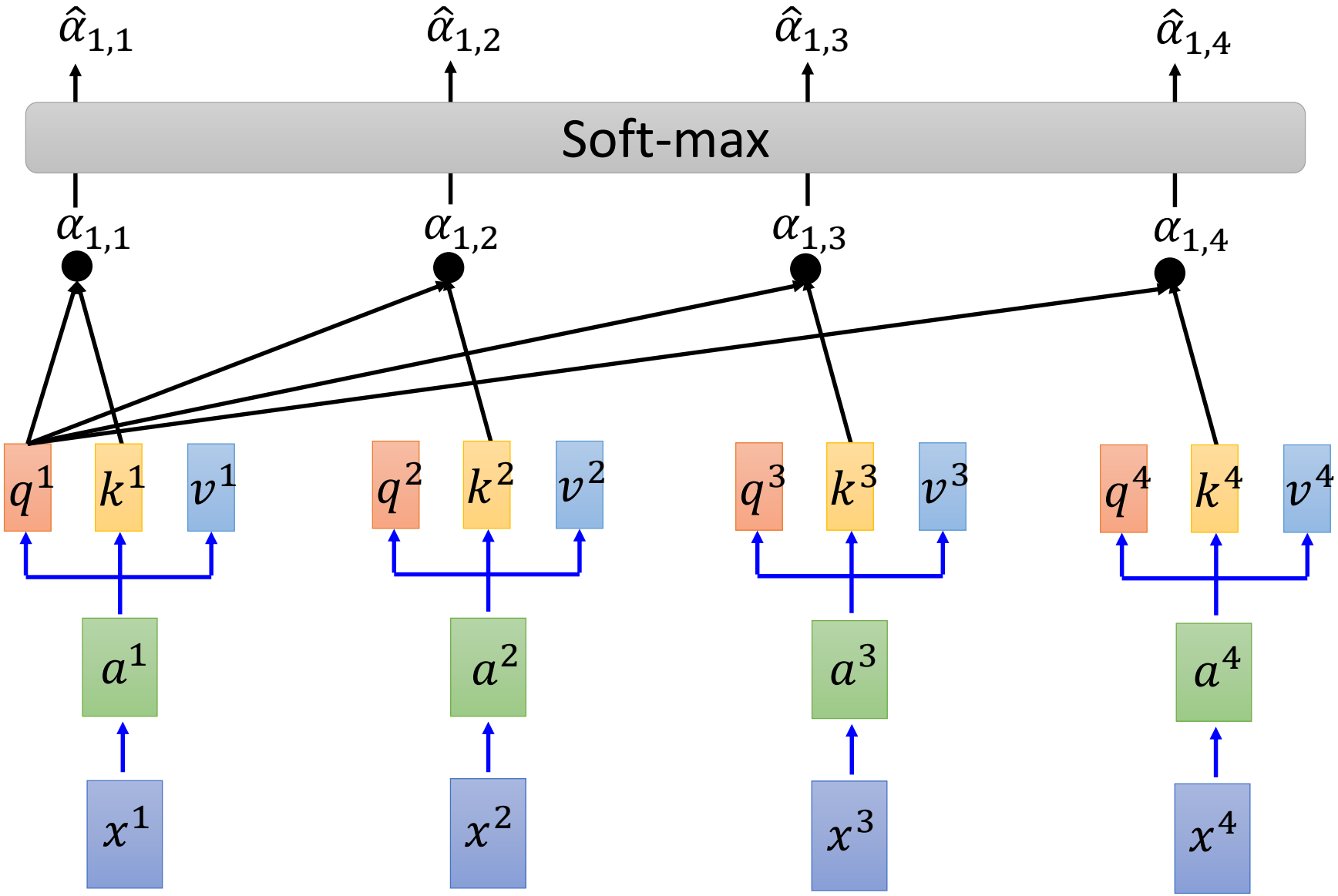
Scaled Dot-Product Attention: $\alpha_{1,i} = q^1 \cdot k^i / \sqrt{d}$

dot product



Self-attention

$$\hat{\alpha}_{1,i} = exp(\alpha_{1,i}) / \sum_j exp(\alpha_{1,j})$$

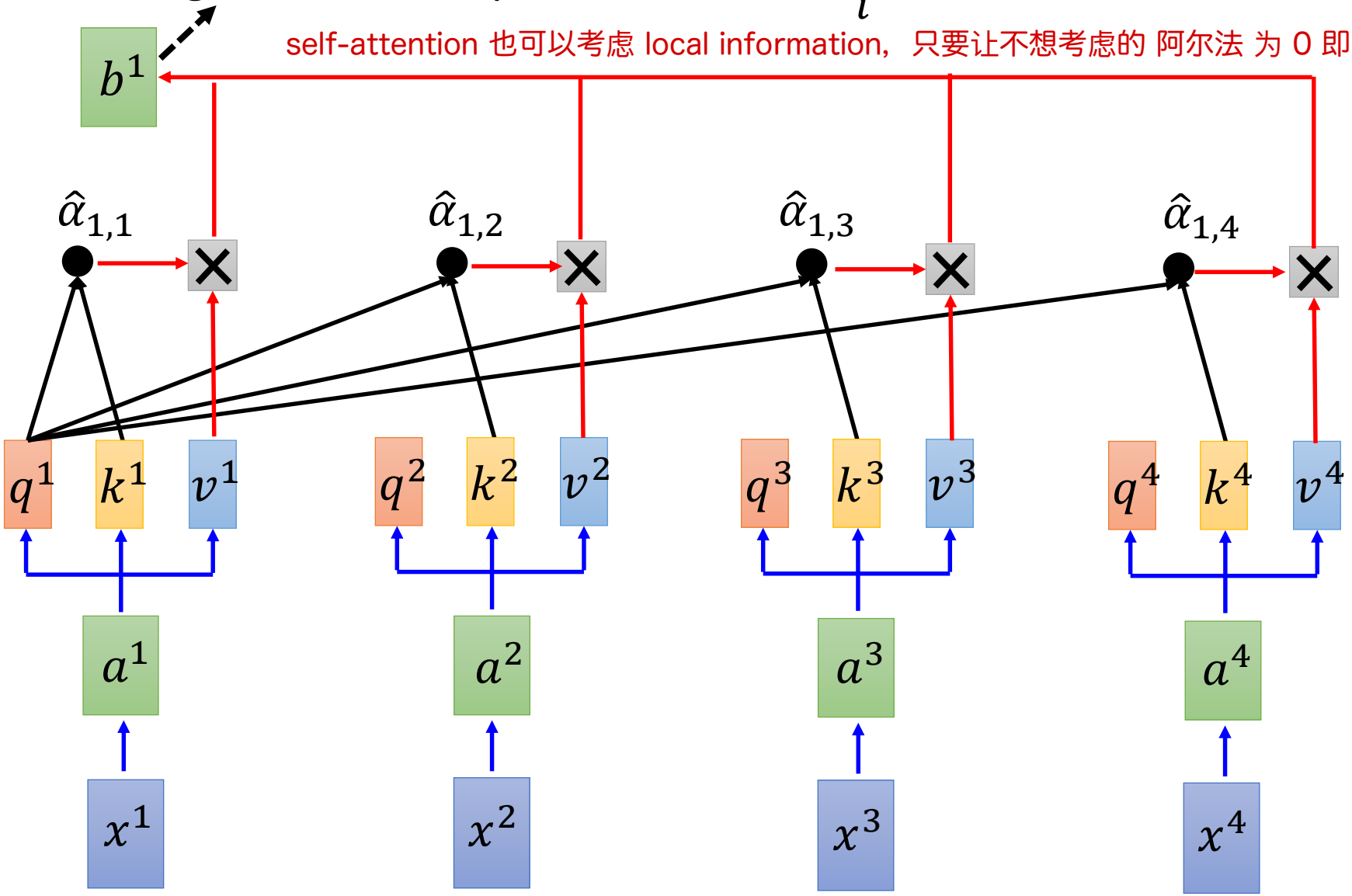


Self-attention

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

Considering the whole sequence

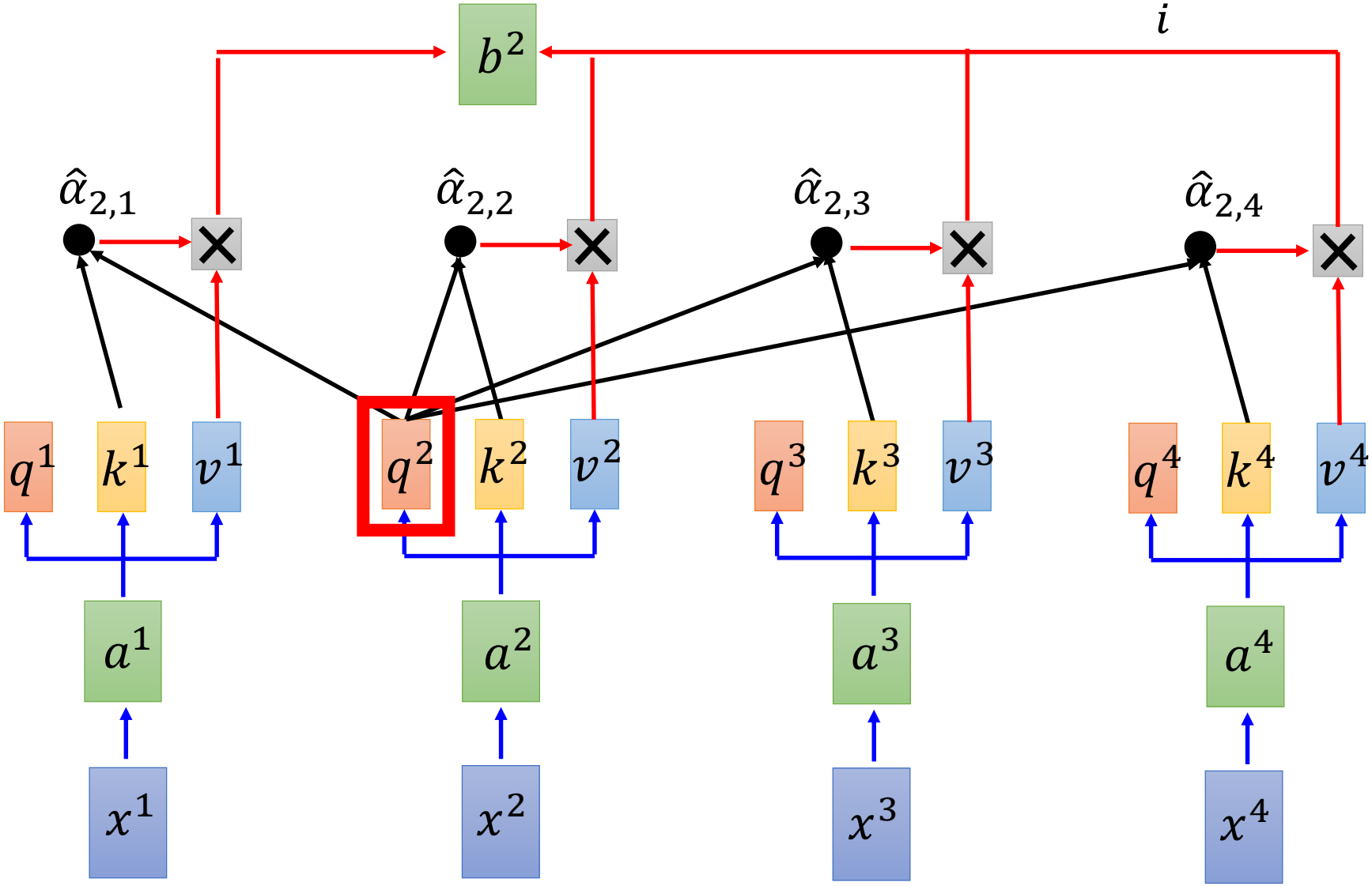
self-attention 也可以考虑 local information，只要让不想考虑的 阿尔法 为 0 即可



Self-attention

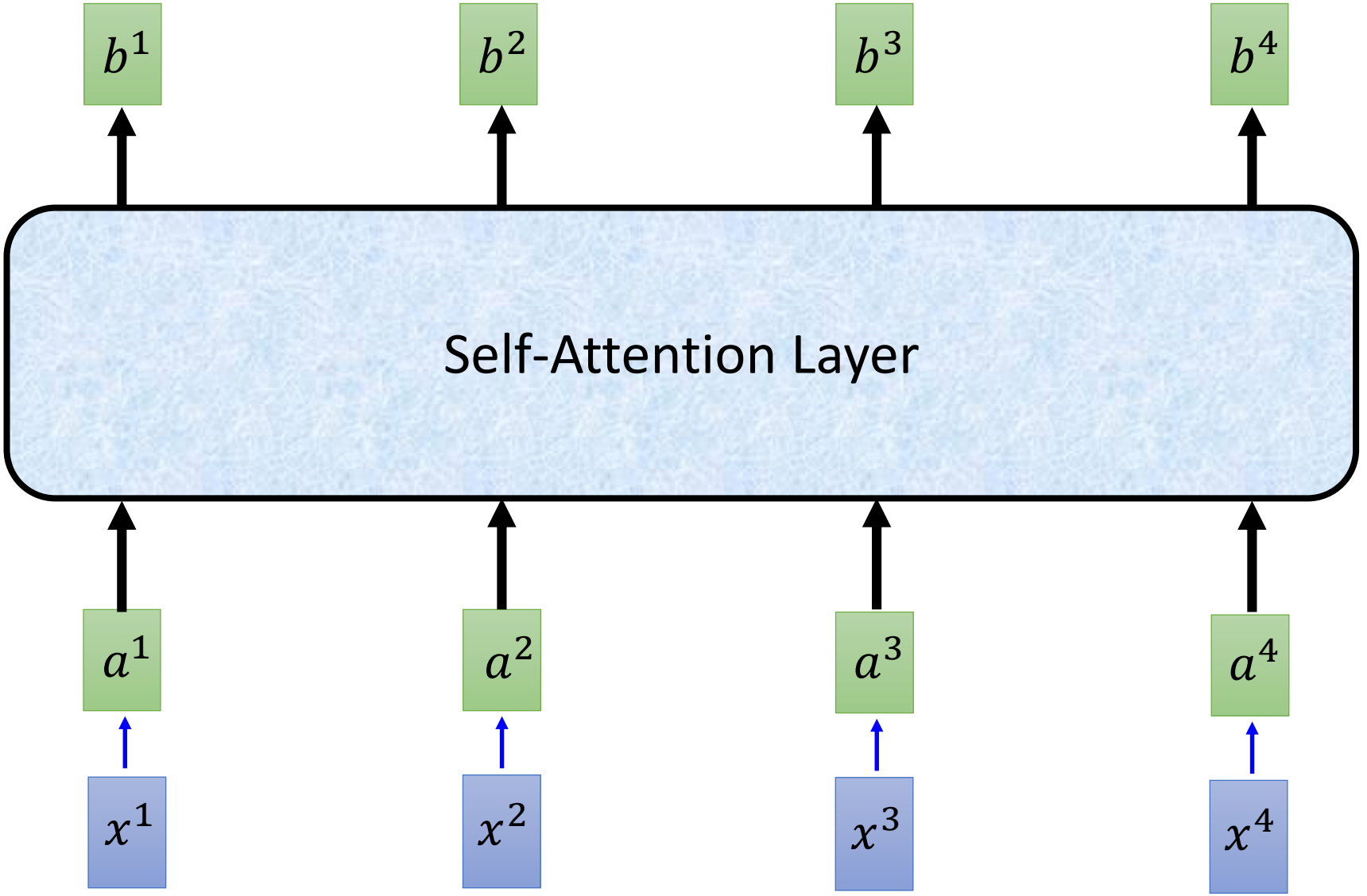
拿每個 query q 去對每個 key k 做 attention

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



Self-attention

b^1, b^2, b^3, b^4 can be parallelly computed.



Self-attention

用矩阵运算来表示

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

$$k^i = W^k a^i$$

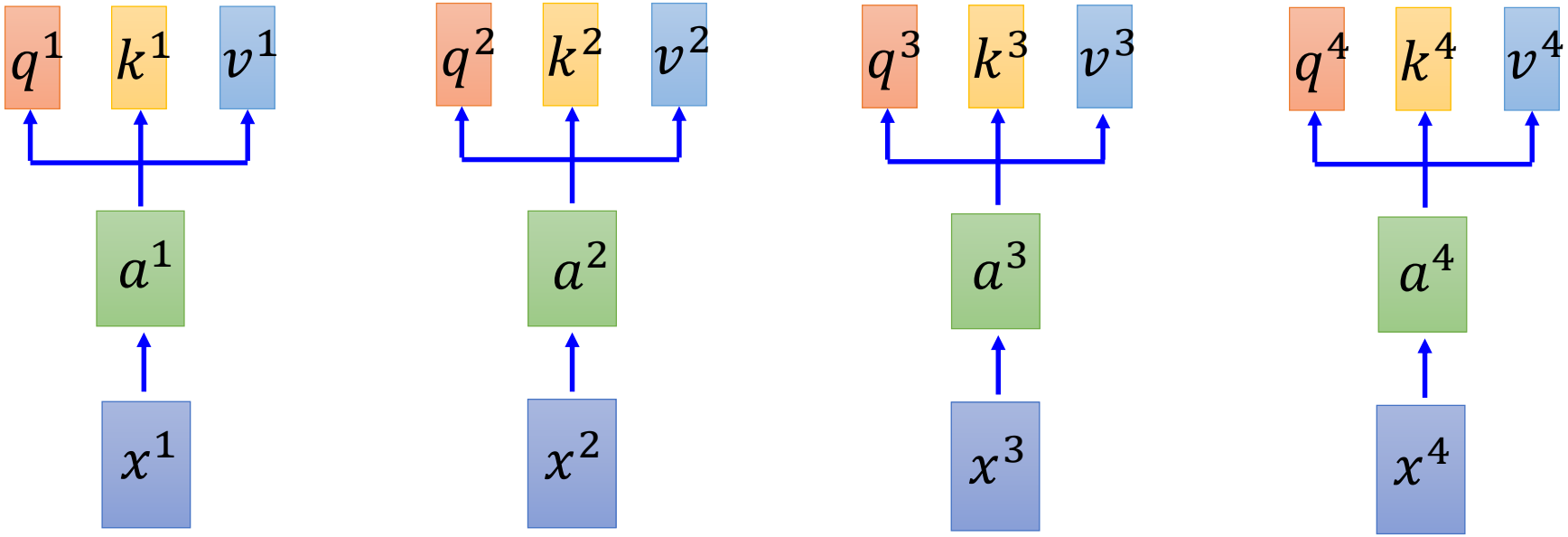
$$v^i = W^v a^i$$

①

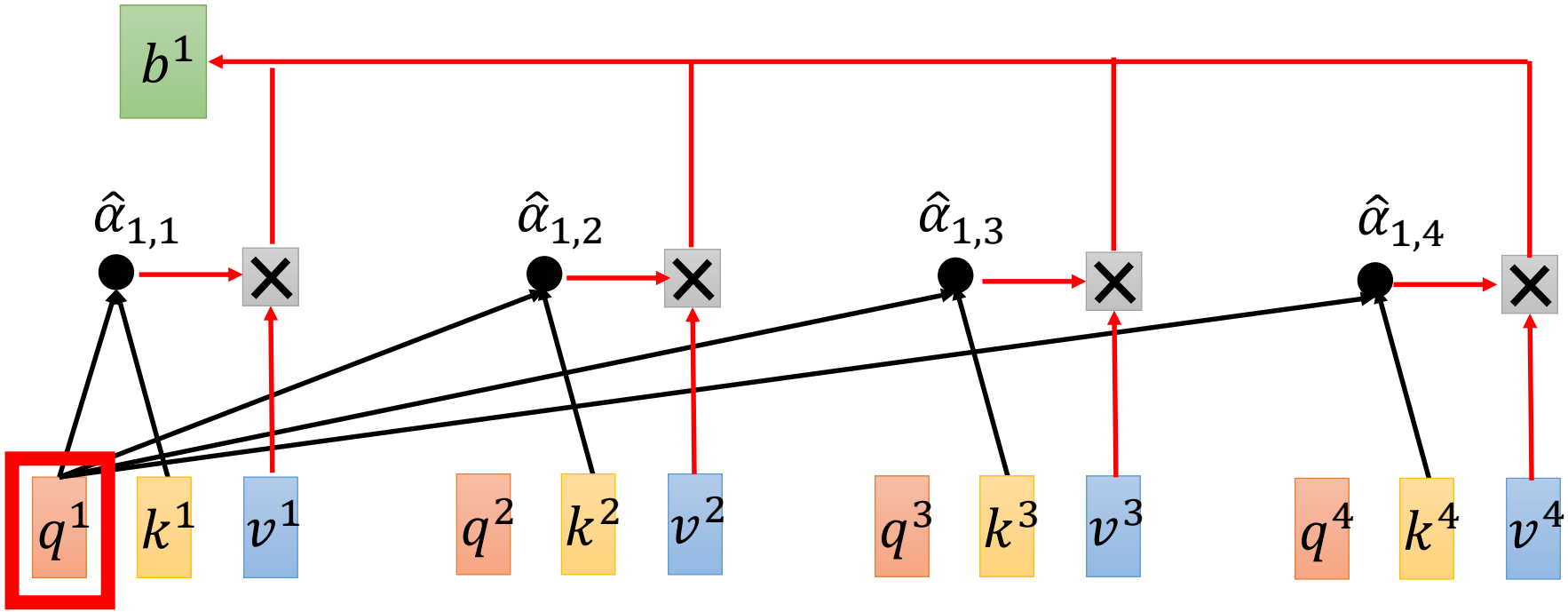
$$\underbrace{q^1\ q^2\ q^3\ q^4}_{Q} = W^q \underbrace{a^1\ a^2\ a^3\ a^4}_{I}$$

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

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



Self-attention



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

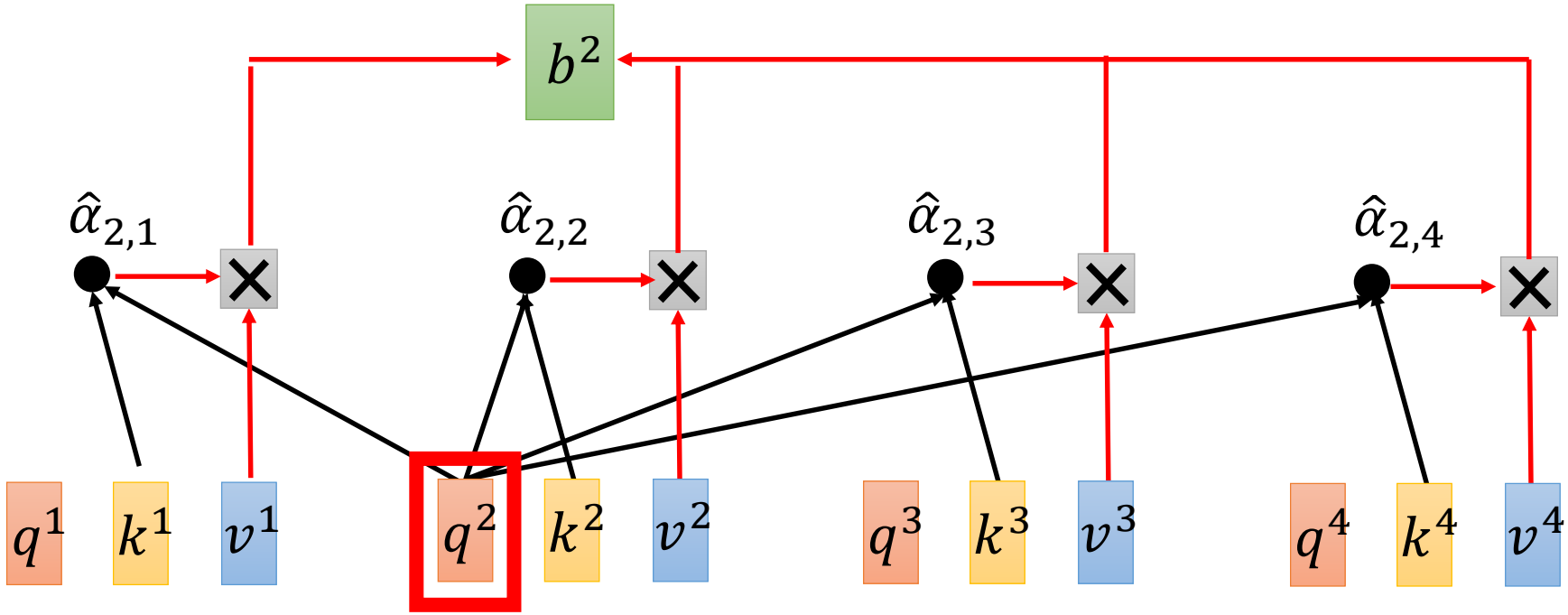
$\alpha_{1,3} = k^3 \; q^1$ $\alpha_{1,4} = k^4 \; q^1$

(ignore $\sqrt{d}$ for simplicity)

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

Self-attention

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

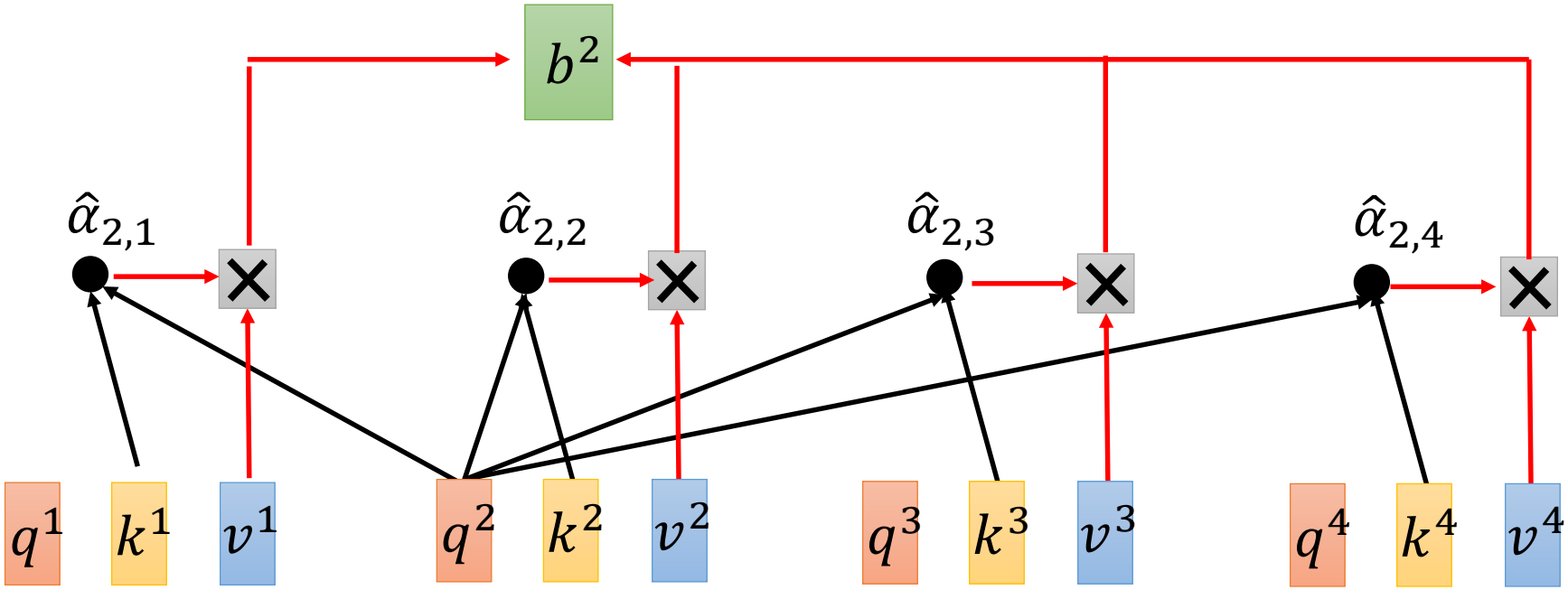


$$\hat{A} \quad \textcircled{4} \quad A \quad \textcircled{3} \quad = \quad K^T \quad Q$$

$$\begin{bmatrix} \hat{\alpha}_{1,1} & \hat{\alpha}_{2,1} & \hat{\alpha}_{3,1} & \hat{\alpha}_{4,1} \\ \hat{\alpha}_{1,2} & \hat{\alpha}_{2,2} & \hat{\alpha}_{3,2} & \hat{\alpha}_{4,2} \\ \hat{\alpha}_{1,3} & \hat{\alpha}_{2,3} & \hat{\alpha}_{3,3} & \hat{\alpha}_{4,3} \\ \hat{\alpha}_{1,4} & \hat{\alpha}_{2,4} & \hat{\alpha}_{3,4} & \hat{\alpha}_{4,4} \end{bmatrix} \leftarrow \begin{bmatrix} \alpha_{1,1} & \alpha_{2,1} & \alpha_{3,1} & \alpha_{4,1} \\ \alpha_{1,2} & \alpha_{2,2} & \alpha_{3,2} & \alpha_{4,2} \\ \alpha_{1,3} & \alpha_{2,3} & \alpha_{3,3} & \alpha_{4,3} \\ \alpha_{1,4} & \alpha_{2,4} & \alpha_{3,4} & \alpha_{4,4} \end{bmatrix} = \begin{bmatrix} k^1 \\ k^2 \\ k^3 \\ k^4 \end{bmatrix} \begin{bmatrix} q^1 & q^2 & q^3 & q^4 \end{bmatrix}$$

Self-attention

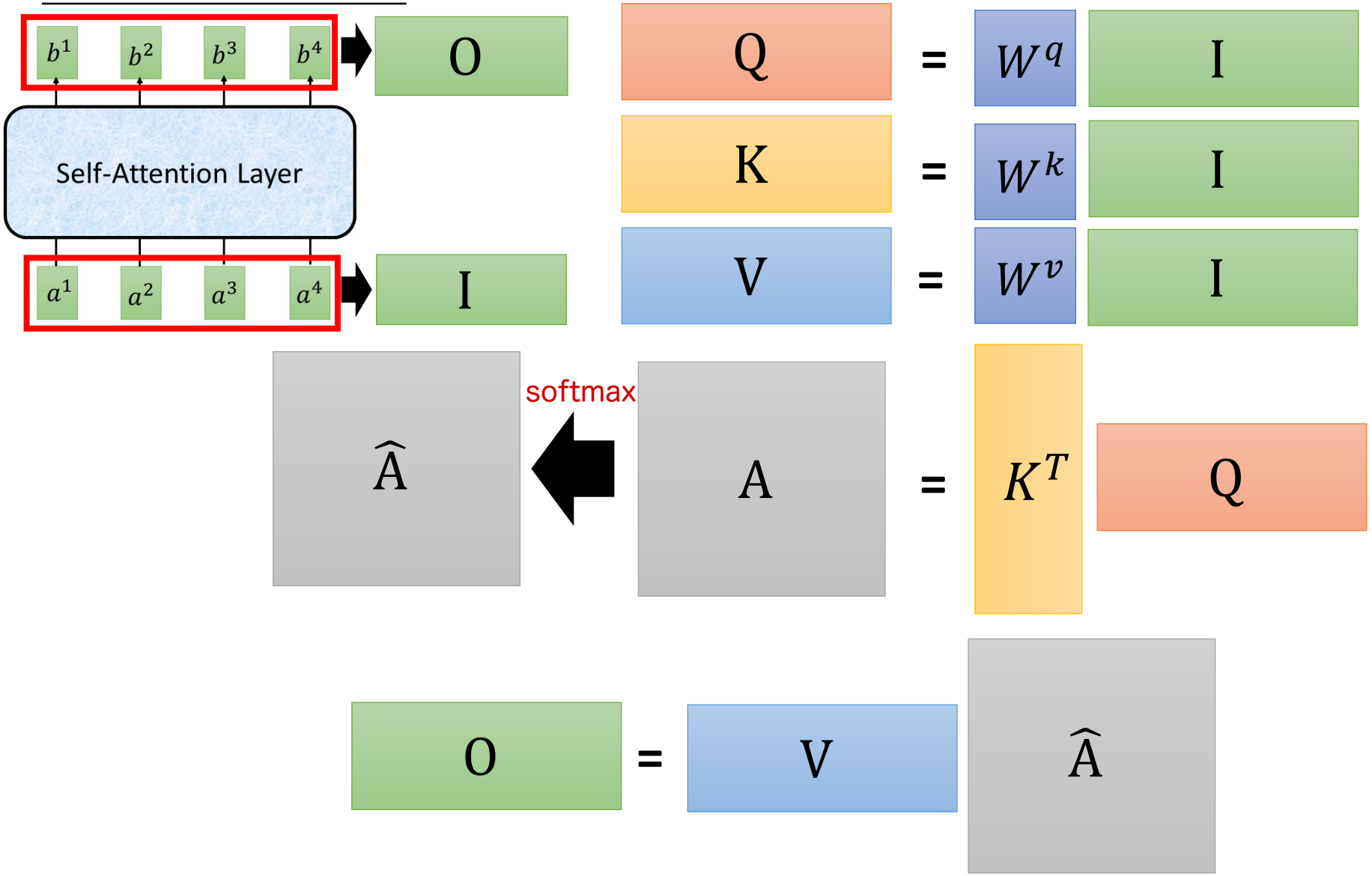
$$b^2 = \sum_i \hat{\alpha}_{2,i} v^i$$



$$\begin{bmatrix} b^1 & b^2 & b^3 & b^4 \end{bmatrix} = \begin{bmatrix} v^1 & v^2 & v^3 & v^4 \end{bmatrix} \begin{bmatrix} \hat{\alpha}_{1,1} & \hat{\alpha}_{2,1} & \hat{\alpha}_{3,1} & \hat{\alpha}_{4,1} \\ \hat{\alpha}_{1,2} & \hat{\alpha}_{2,2} & \hat{\alpha}_{3,2} & \hat{\alpha}_{4,2} \\ \hat{\alpha}_{1,3} & \hat{\alpha}_{2,3} & \hat{\alpha}_{3,3} & \hat{\alpha}_{4,3} \\ \hat{\alpha}_{1,4} & \hat{\alpha}_{2,4} & \hat{\alpha}_{3,4} & \hat{\alpha}_{4,4} \end{bmatrix}$$

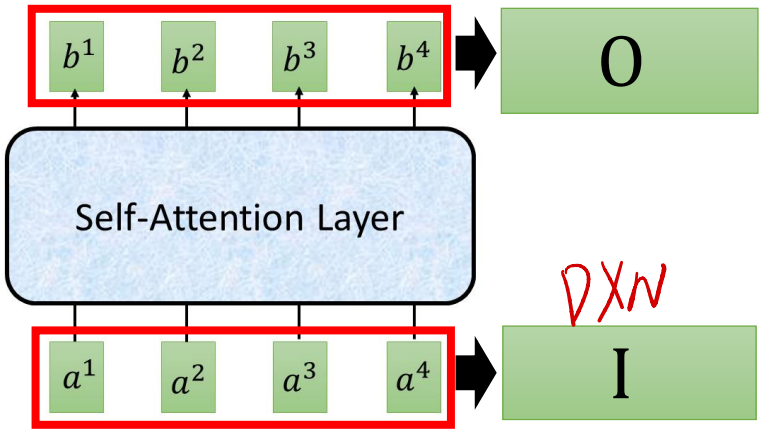
O = V ⑤ $\hat{A}$

Self-attention (整个流程)



反正就是一堆矩陣乘法，用 GPU 可以加速

Self-attention （整个流程）



$$O = \hat{A} \cdot V$$

$$A = K^T \cdot Q$$

$$\hat{A} = \mathrm{softmax}(A)$$

$$Q = W^q \cdot I$$

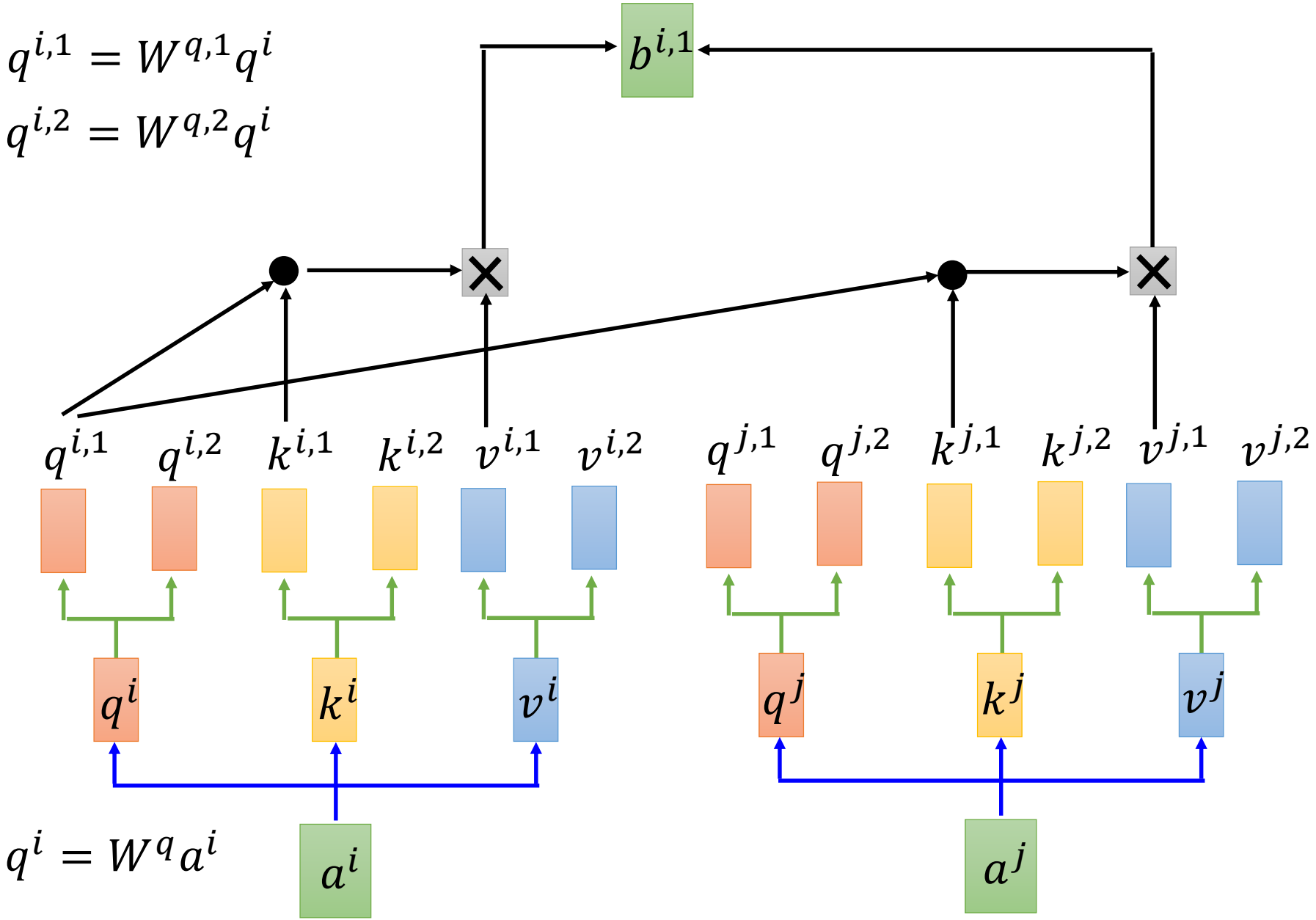
$$K = W^k \cdot I$$

$$V = W^v \cdot I$$

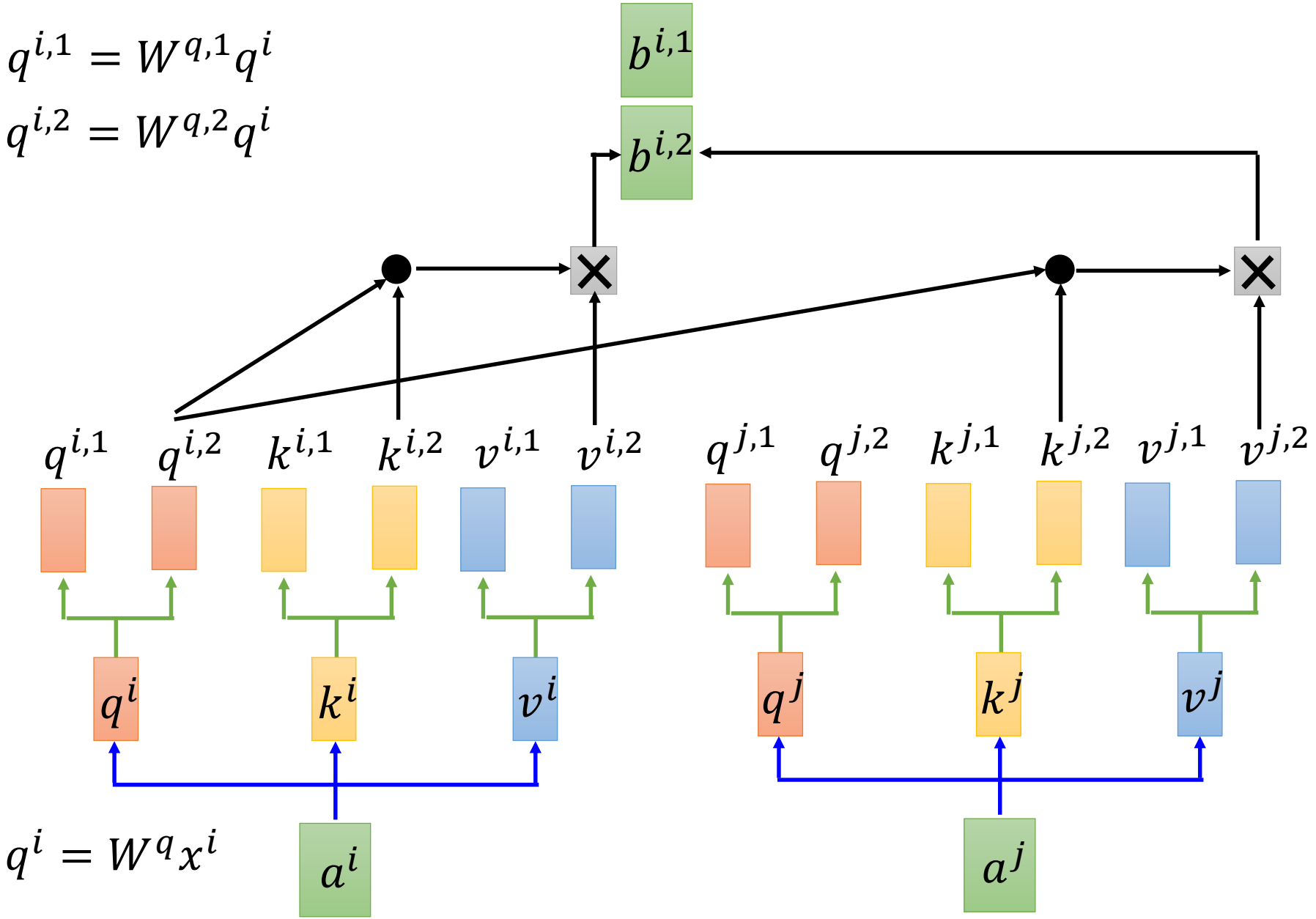
D：每个word被embedding的维度

反正就是一堆矩阵乘法，用 GPU 可以加速

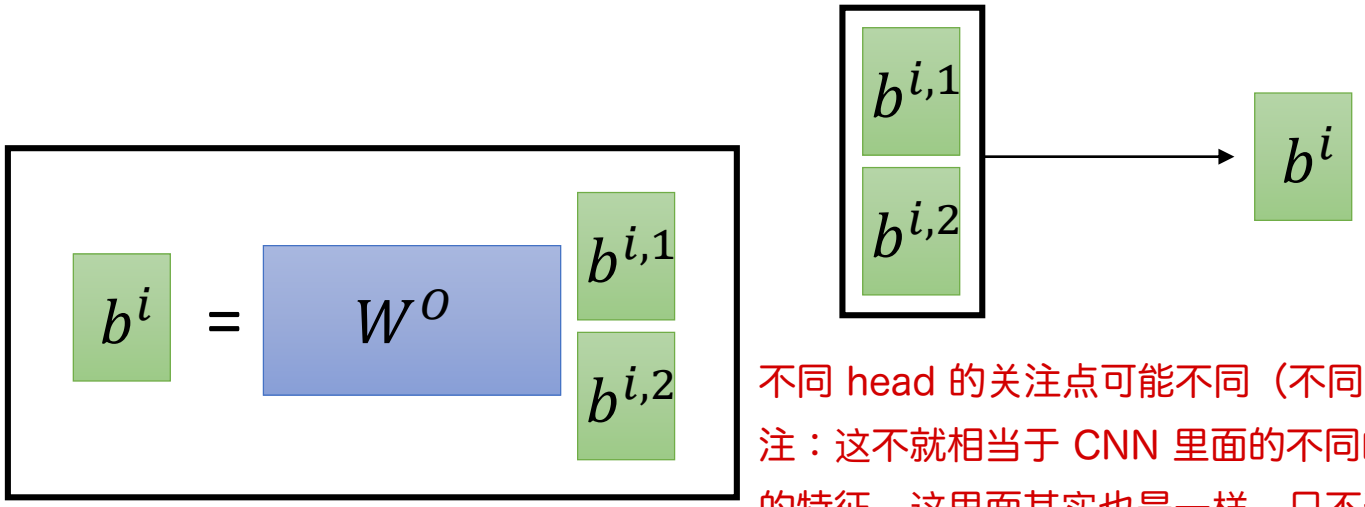
Multi-head Self-attention (2 heads as example)



Multi-head Self-attention (2 heads as example)

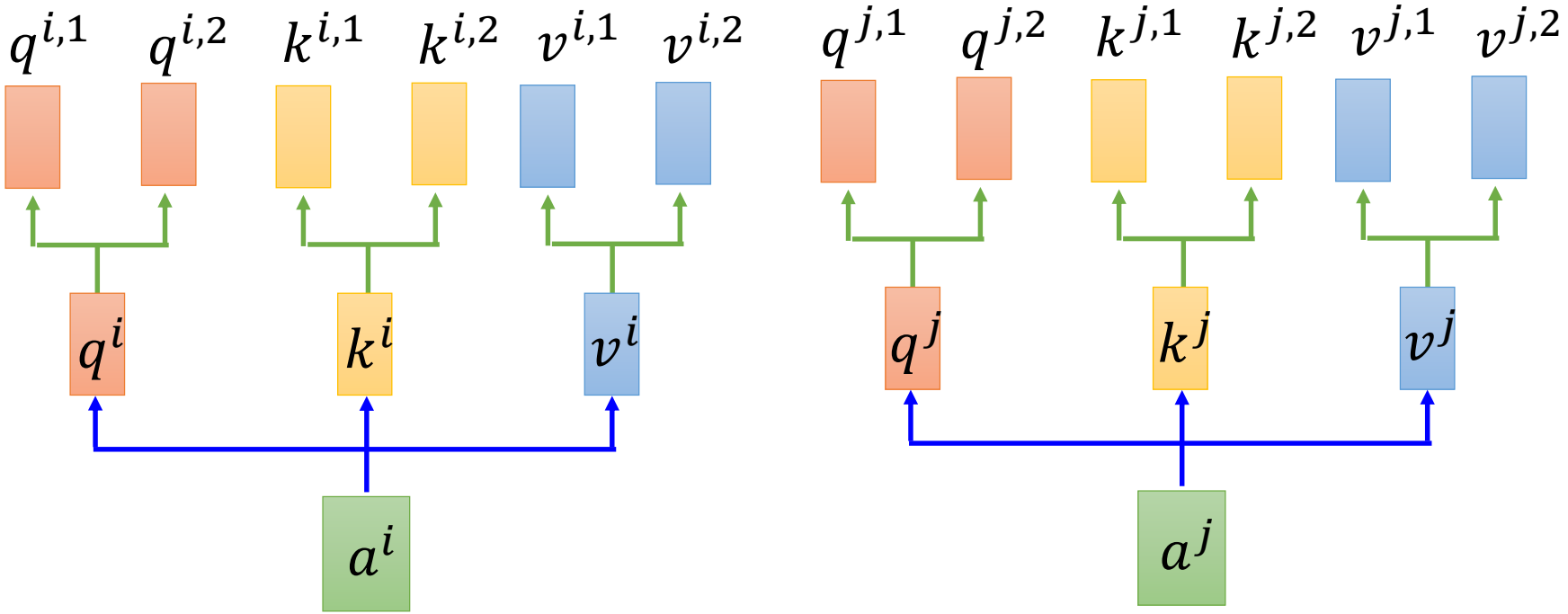


Multi-head Self-attention (2 heads as example)



不同 head 的关注点可能不同（不同的 head，提取不同的特征）

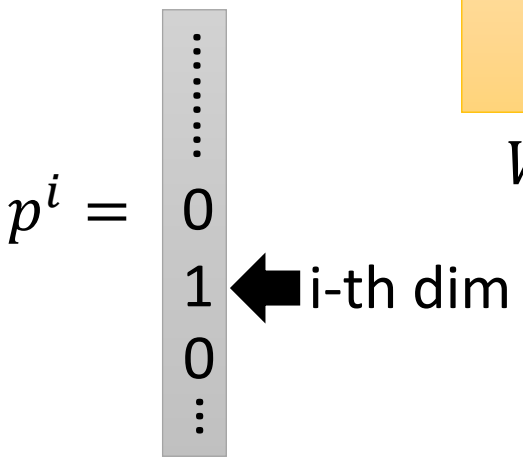
注：这不就相当于 CNN 里面的不同的卷积核吗，用来提取不同的特征。这里面其实也是一样。只不过换了个表达。



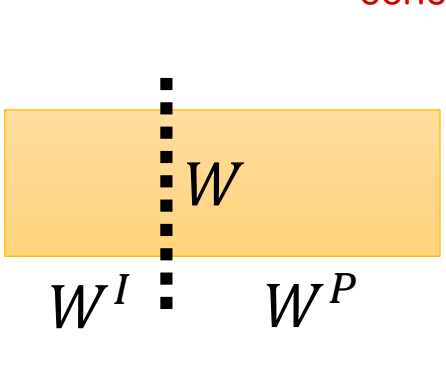
Positional Encoding

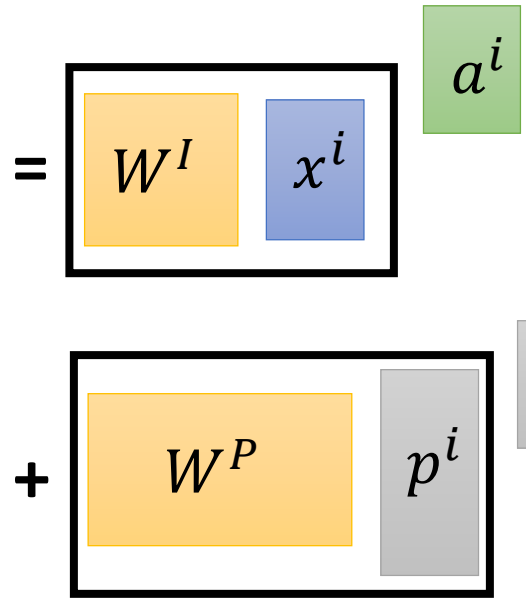
input sequence 的顺序没有考虑进去

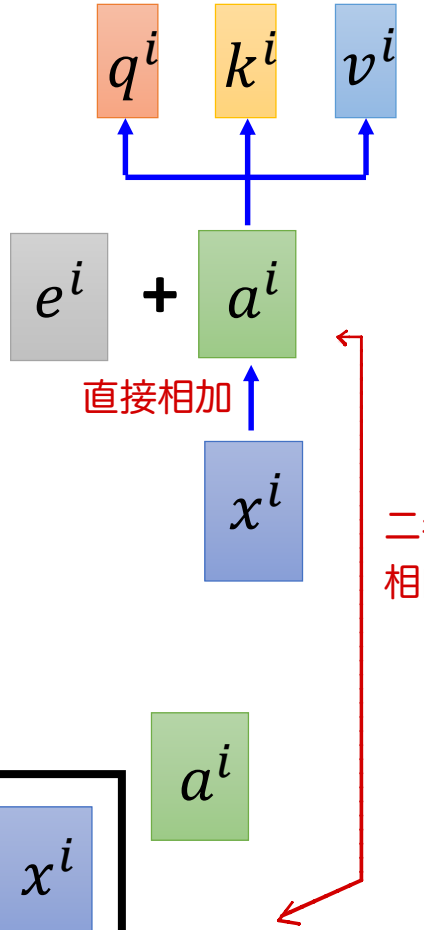
- No position information in self-attention.
- Original paper: each position has a unique positional vector e^i (not learned from data)
- In other words: each x^i appends a one-hot vector p^i



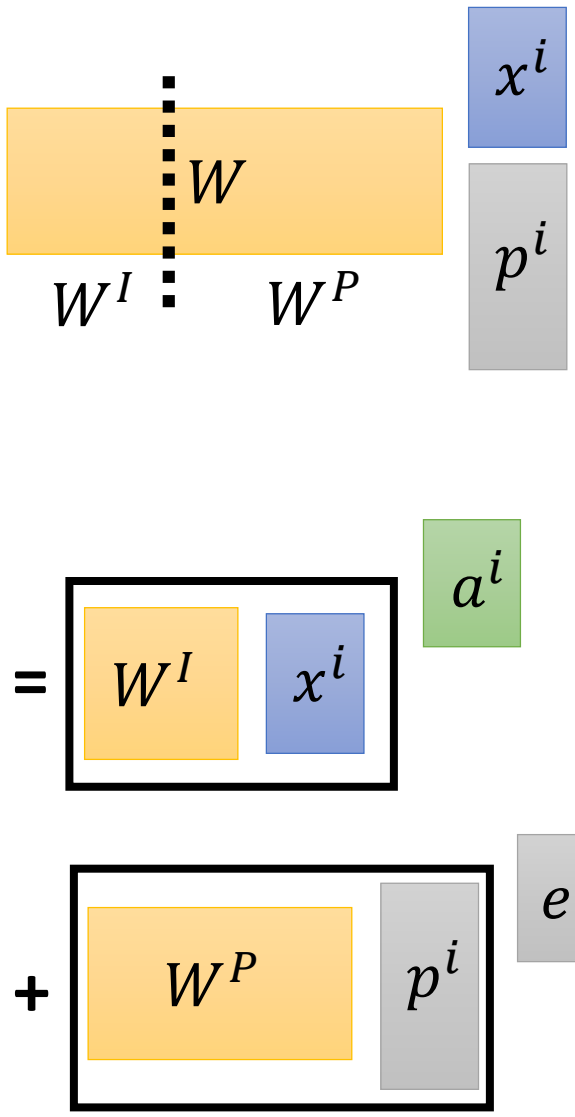
concat



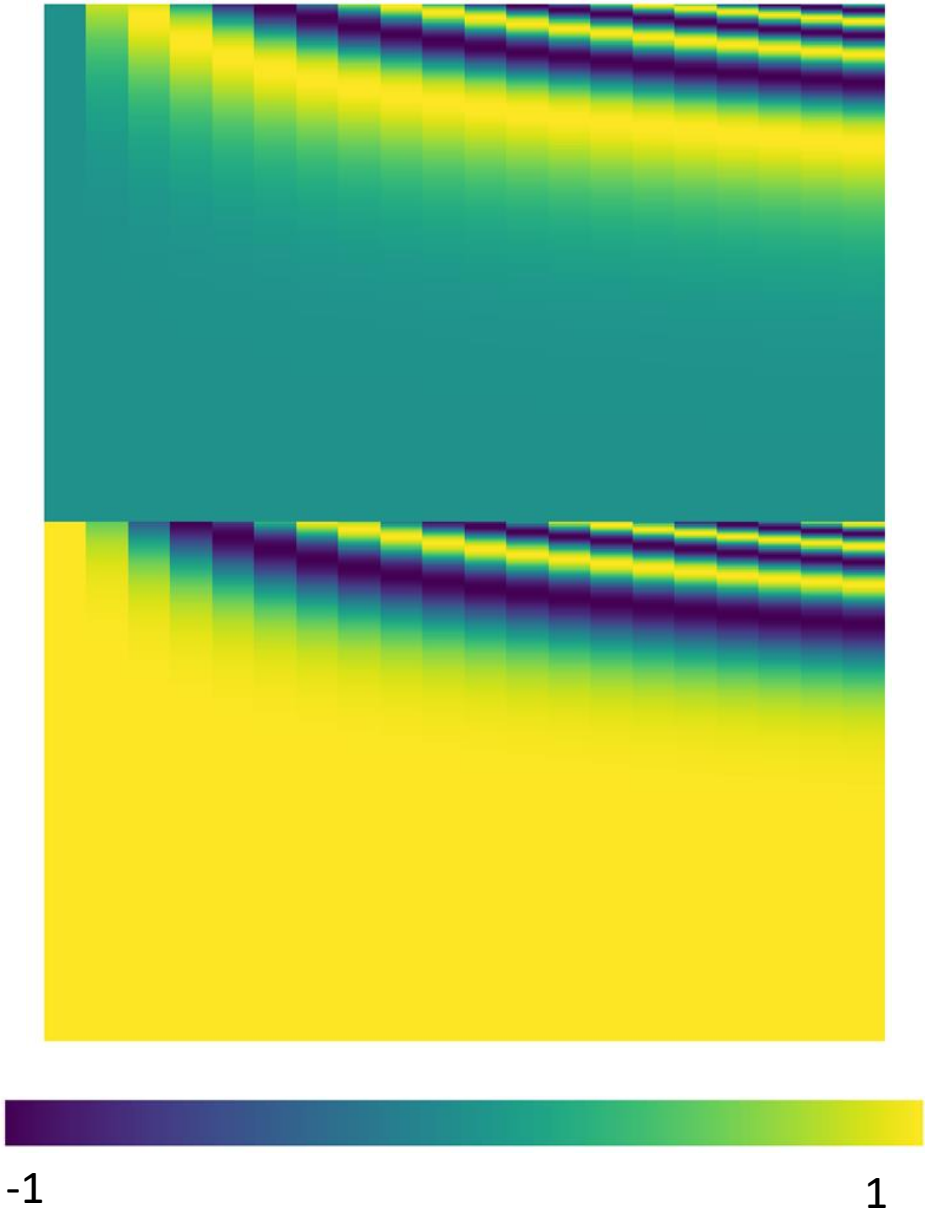




上面的直接相加，等同于在原来 xi 上面 concat 起来



Wp的样子



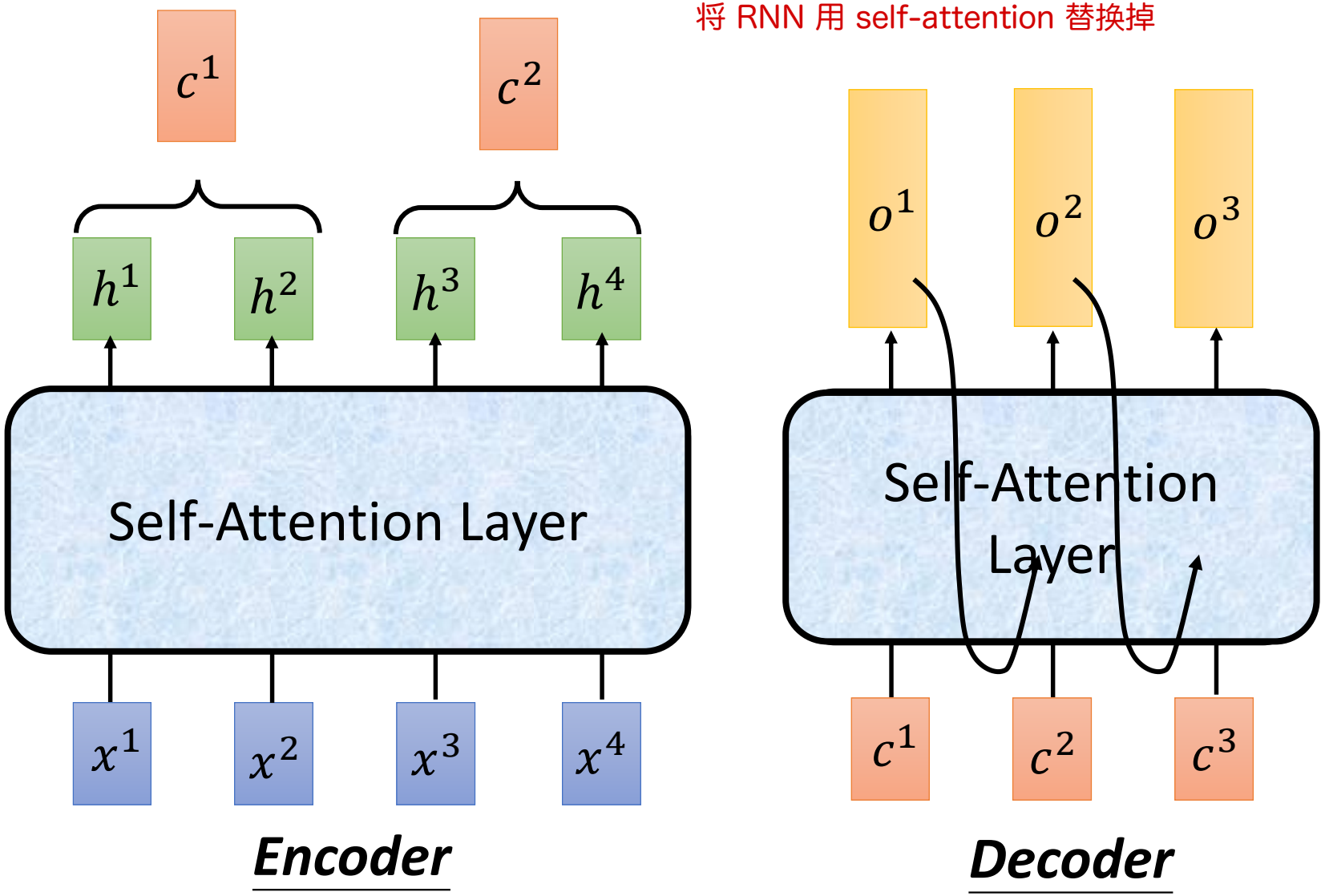
source of image: http://jalammar.github.io/illustrated-transformer/

Review: https://www.youtube.com/watch?v=ZjfjPzXw6og&feature=youtu.be

self-attention 在 seq2seq 的 model 里面是如何被使用的

Seq2seq with Attention

将 RNN 用 self-attention 替换掉



Encoder

Decoder

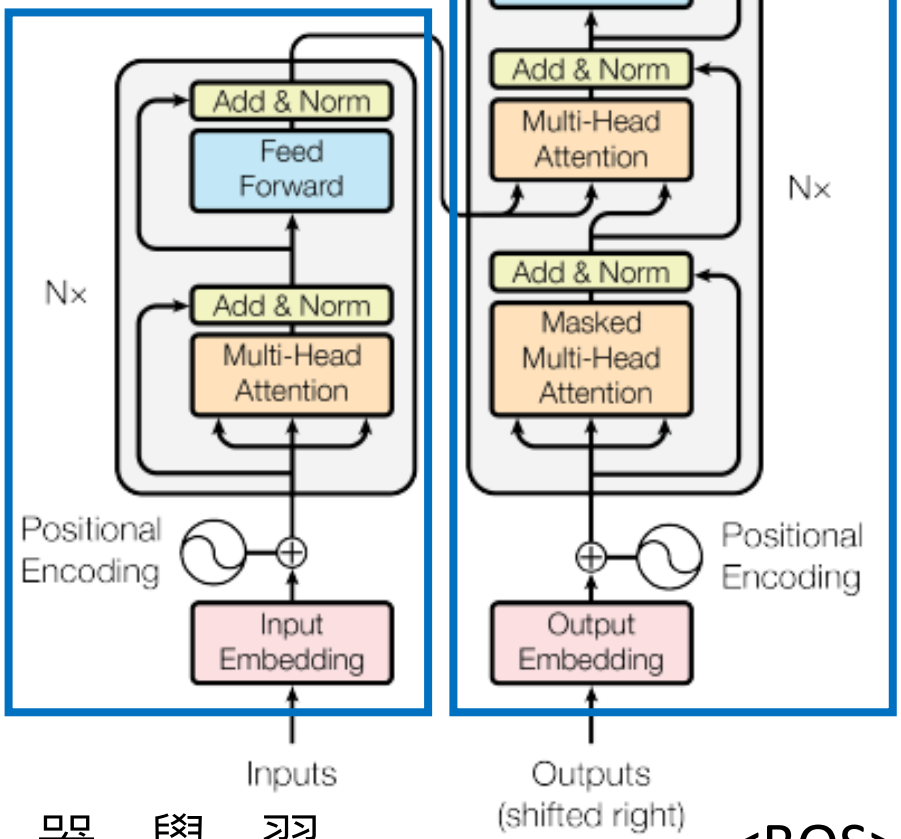
这是个动画

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Transformer

seq2seq model

Using Chinese to English translation as example



Encoder

Decoder

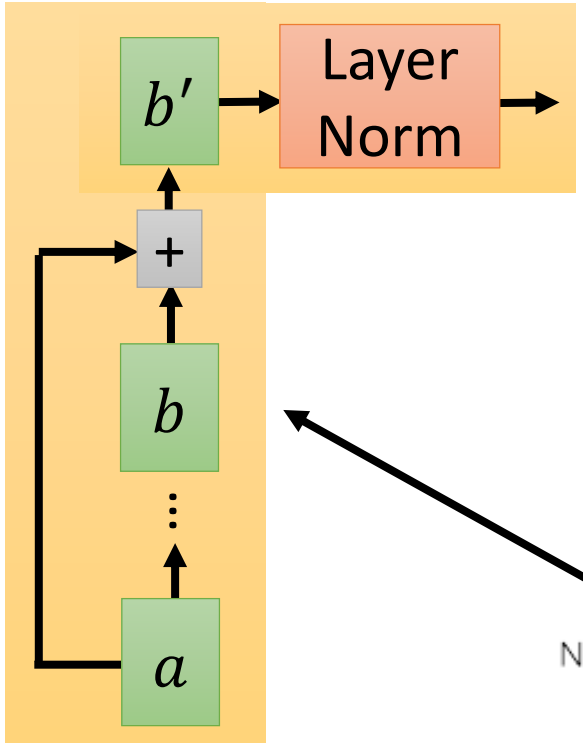
機 器 學 習

<BOS>

machine learning

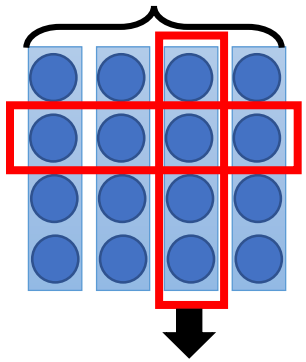
machine

Transformer



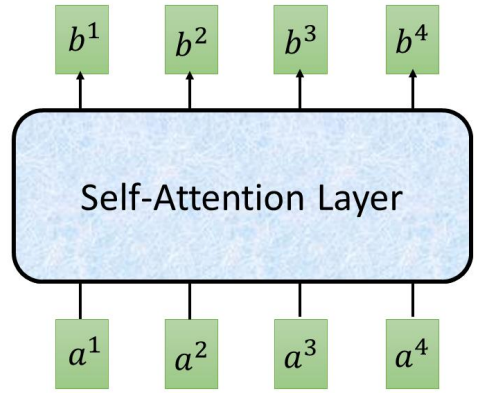
Layer Norm:
https://arxiv.org/abs/1607.06450

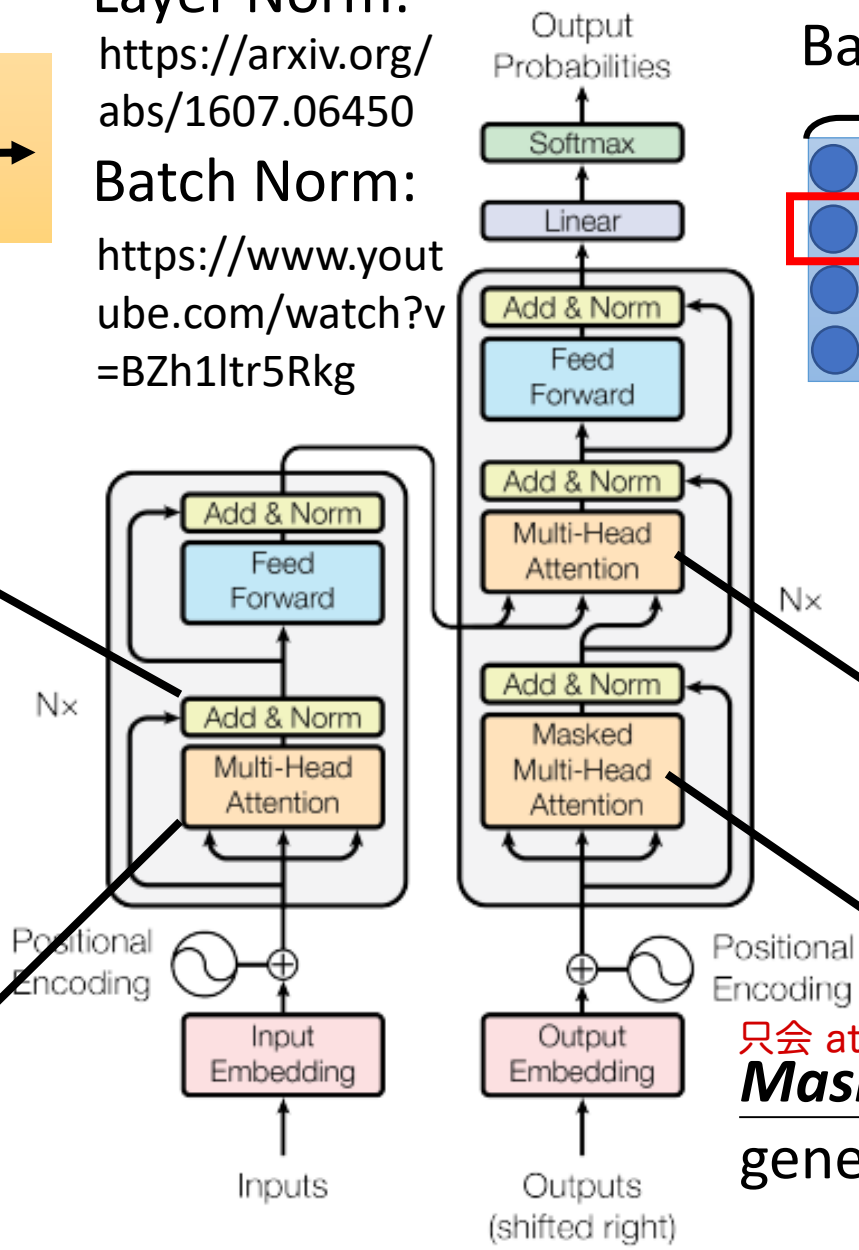
Batch Norm:
https://www.youtube.com/watch?v=BZh1ltr5Rkg



$\mu = 0,$
$\sigma = 1$
Batch

$\mu = 0, \sigma = 1$
Layer





attend on the input sequence

只会 attend 到已经产生出来的部分

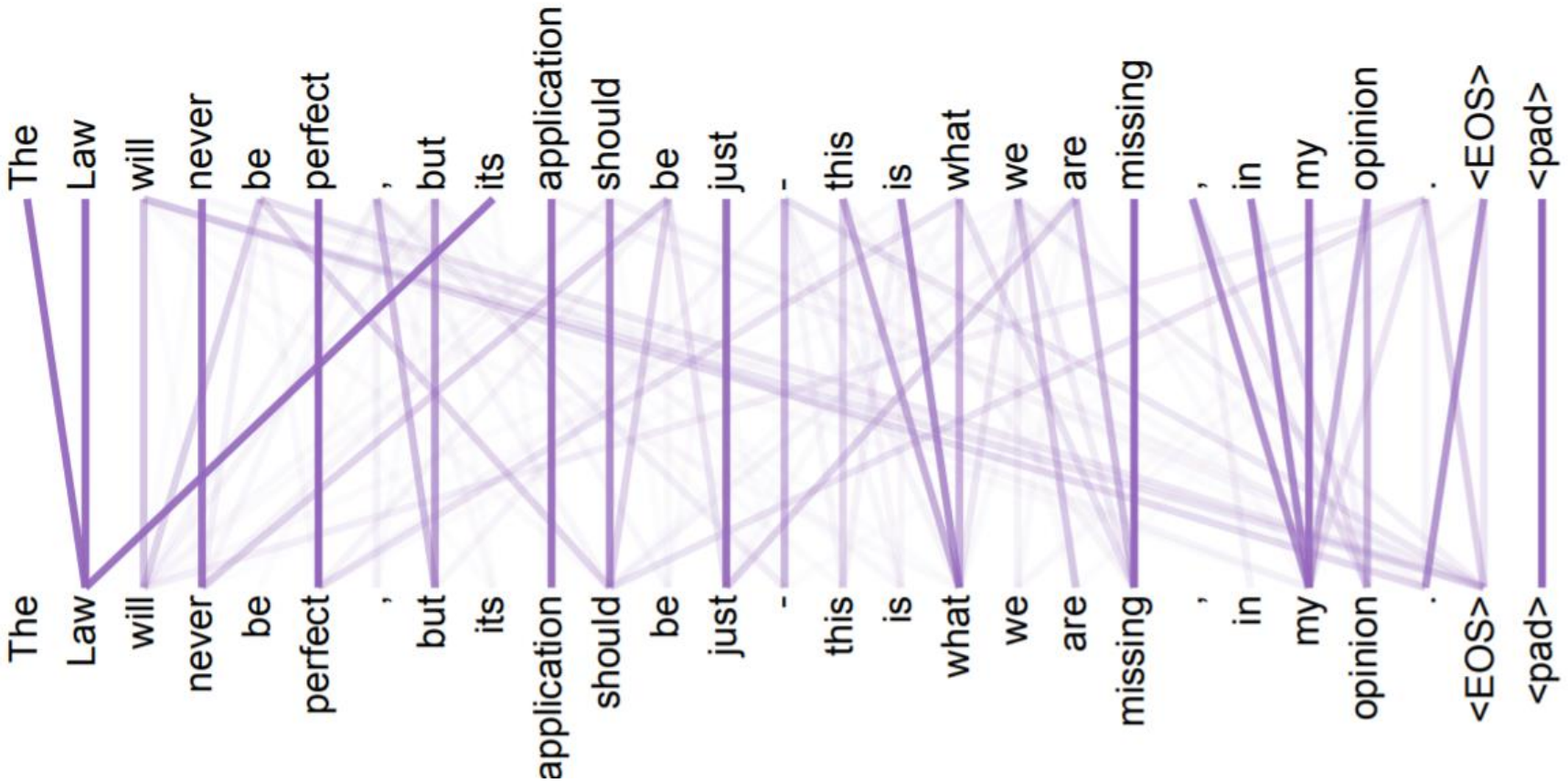
Masked: attend on the generated sequence

BN：同一个batch 里面不同 data 的同一个 dimension 做 normalization

LN：给一个data，在一个 data 的不同 dimension 之间做 normalization

Attention Visualization

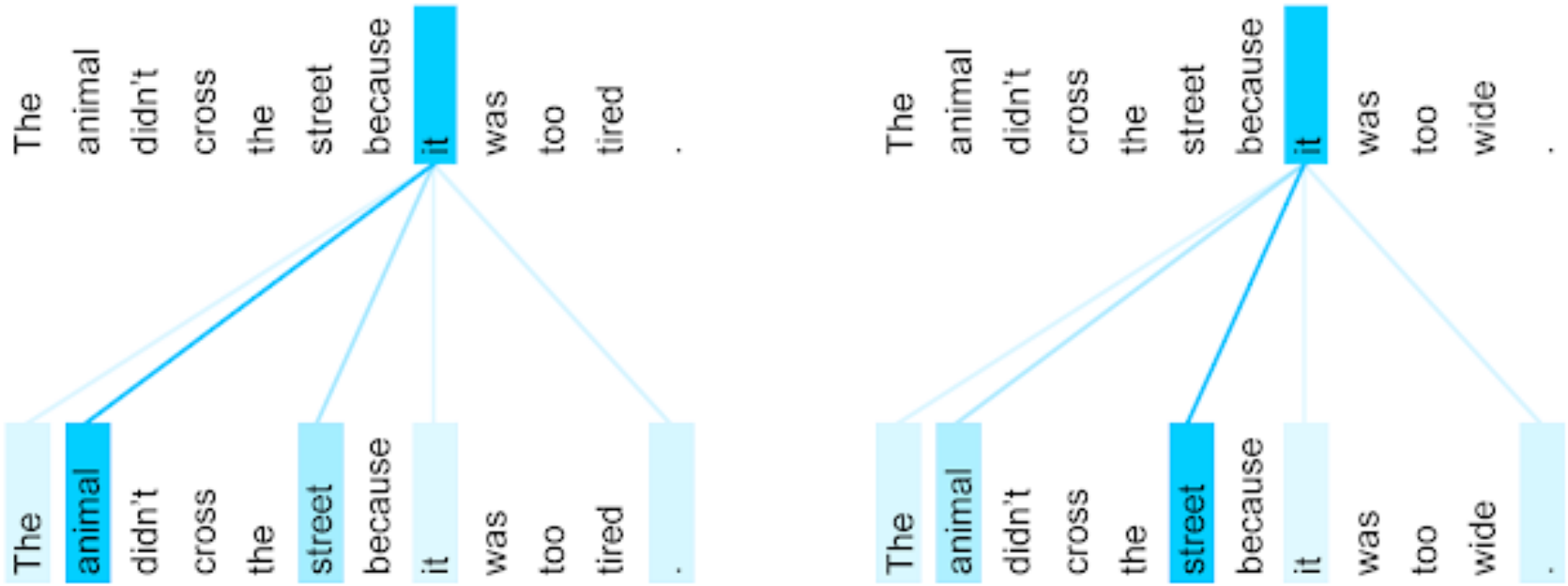
attention weight 越大，线条越粗



https://arxiv.org/abs/1706.03762

Attention Visualization

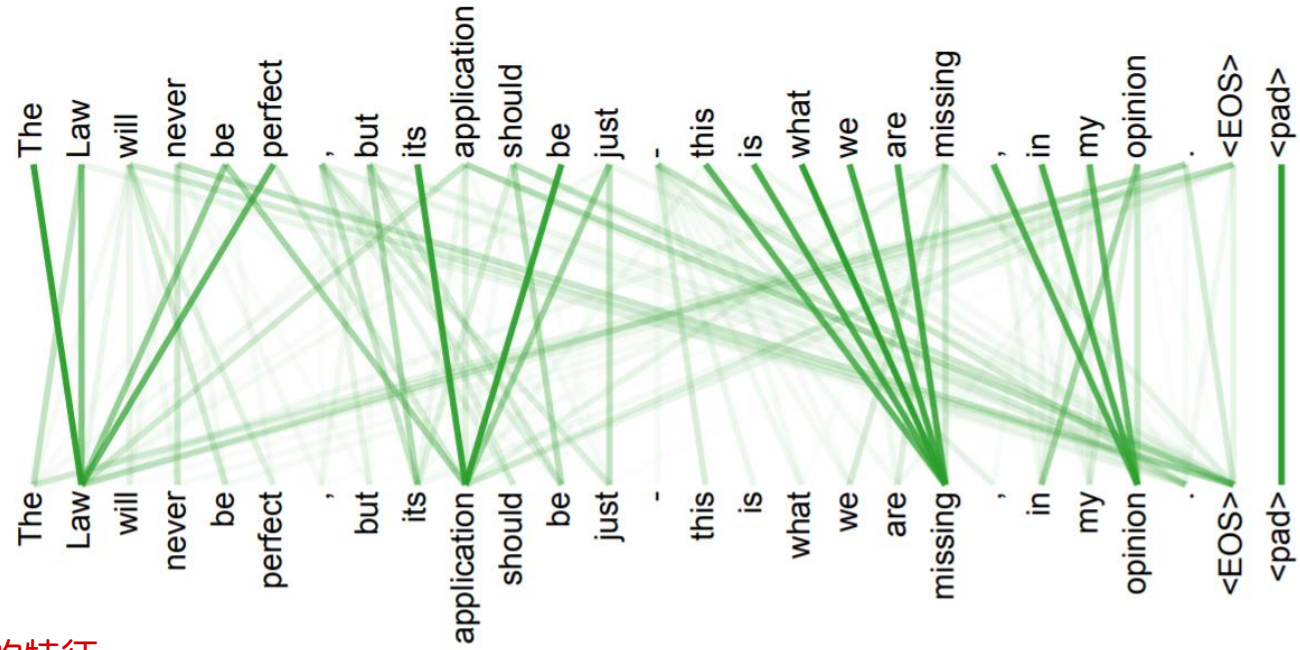
attention weight 越大，线条越粗



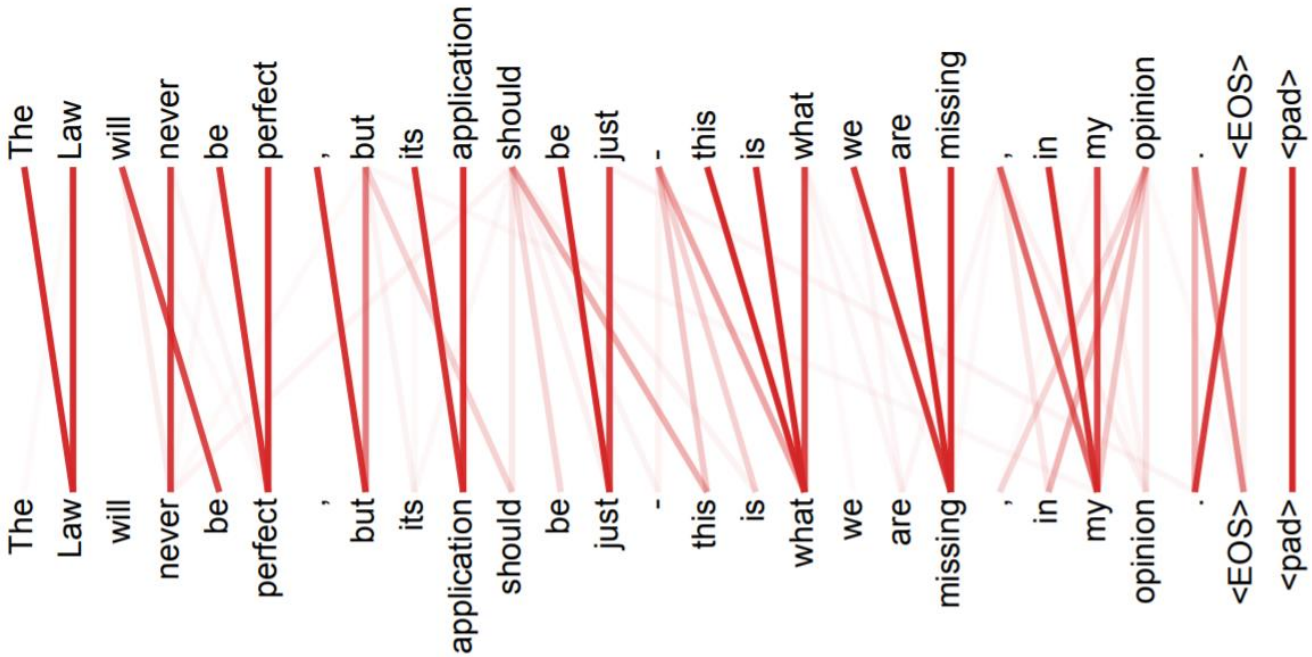
The encoder self-attention distribution for the word “it” from the 5th to the 6th layer of a Transformer trained on English to French translation (one of eight attention heads).

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Multi-head Attention

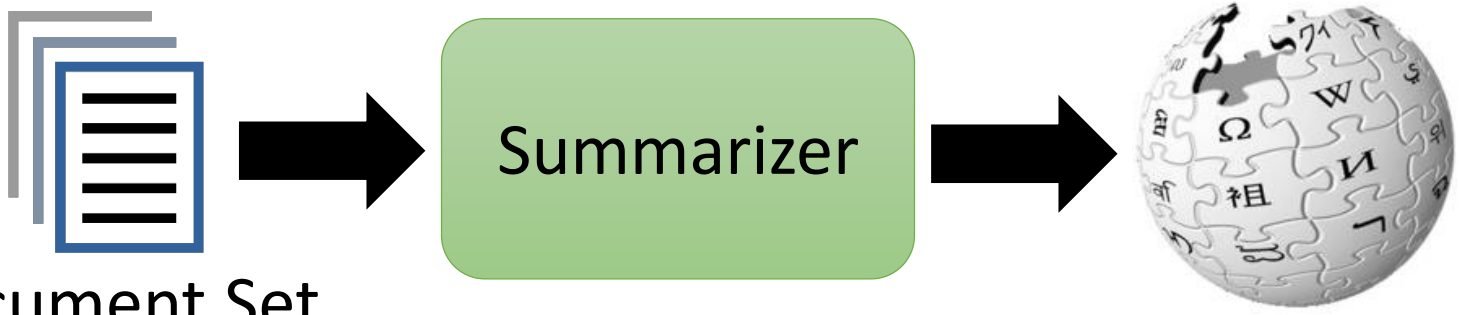


每一组 QKV 在提取不同的特征



Example Application

- If you can use seq2seq, you can use transformer.

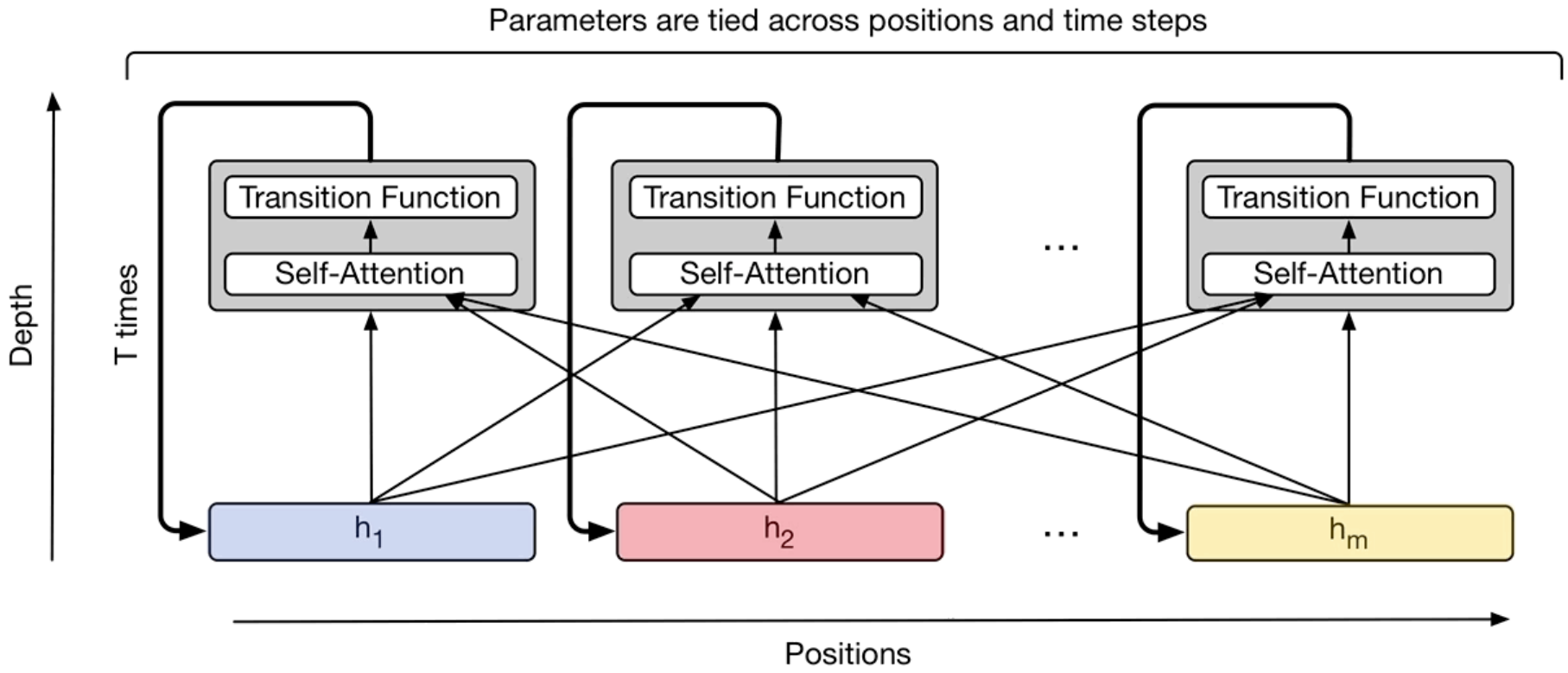


Dataset	Input	Output	# examples
Gigaword (Graff & Cieri, 2003)	10^1	10^1	10^6
CNN/DailyMail (Nallapati et al., 2016)	10^2–10^3	10^1	10^5
WikiSum (ours)	10^2–10^6	10^1–10^3	10^6

https://arxiv.org/abs/1801.10198

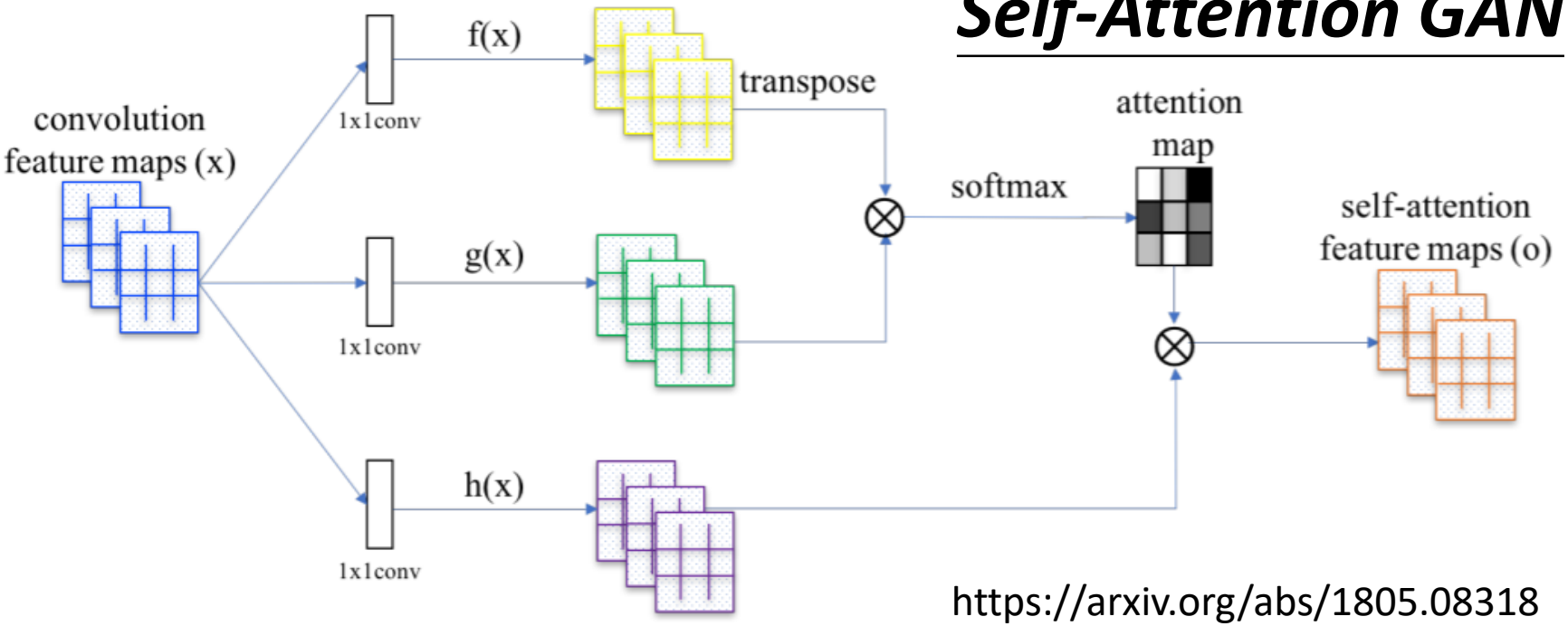
Universal Transformer

（这也是个动画）



https://ai.googleblog.com/2018/08/moving-beyond-translation-with.html

Self-Attention GAN



https://arxiv.org/abs/1805.08318