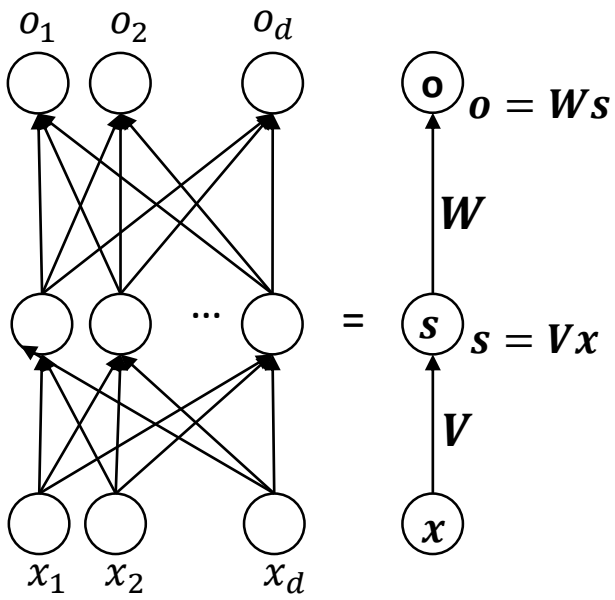
RNN & Transformer

黃志勝（Tommy Huang）
義隆電子 人工智慧研發部
國立陽明交通大學 AI學院 合聘助理教授
國立台北科技大學 電資學院合聘助理教授

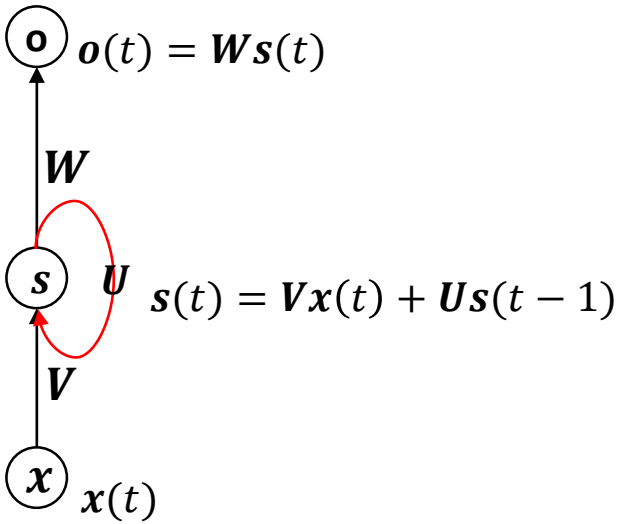
Recurrent Neural Network (RNN)

- RNN和一般MLP的差異就是引進了具有記憶功能的「狀態(State)」

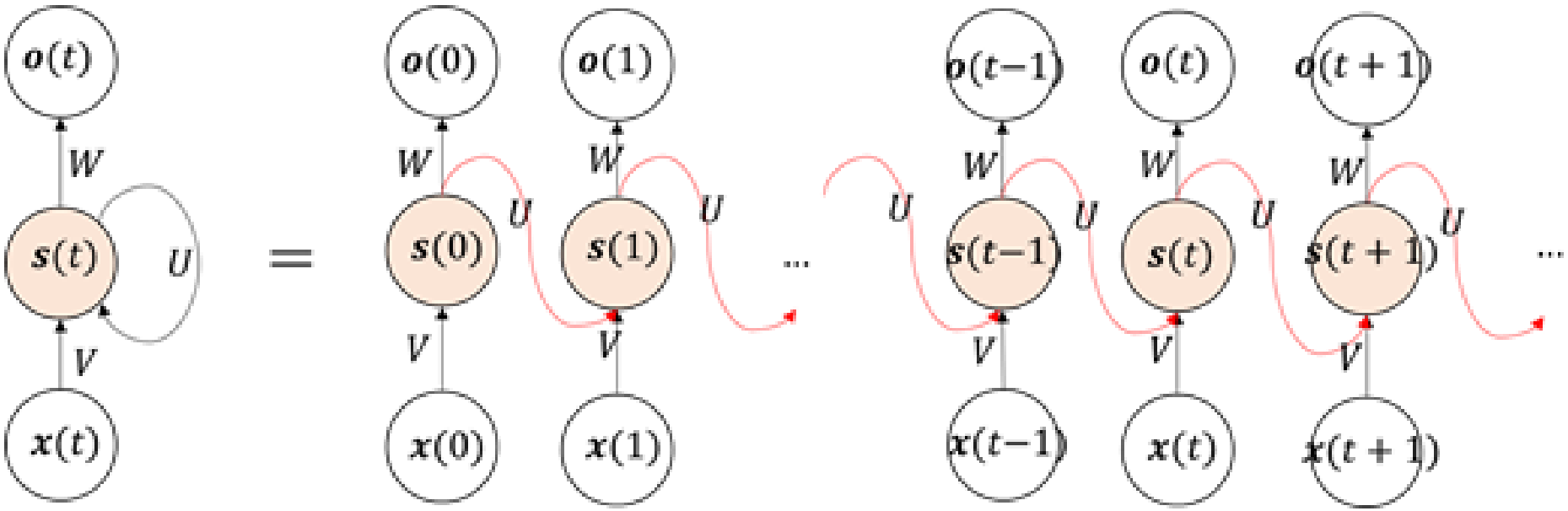
MLP



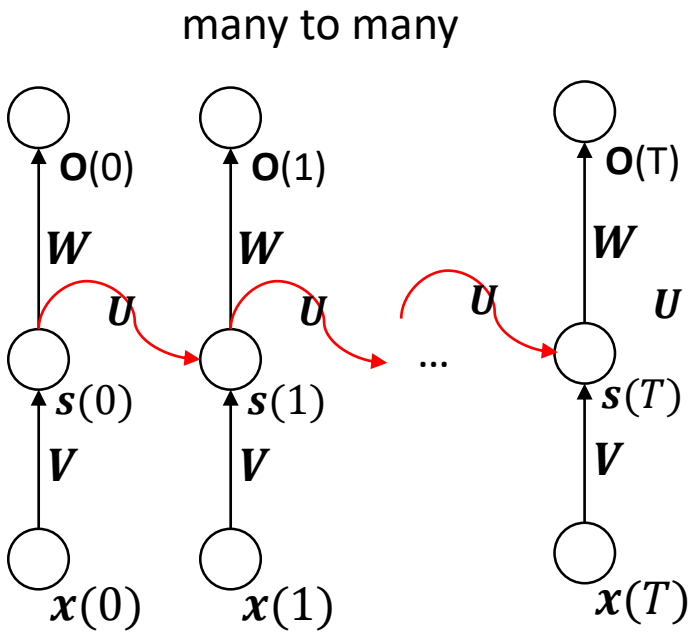
RNN

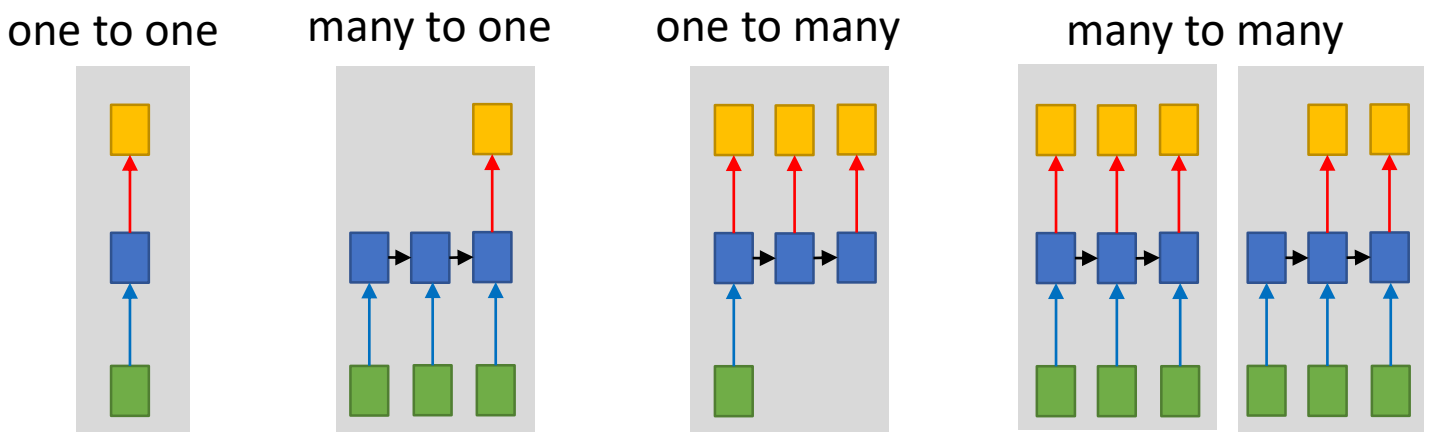


Recurrent Neural Network (RNN)



RNN

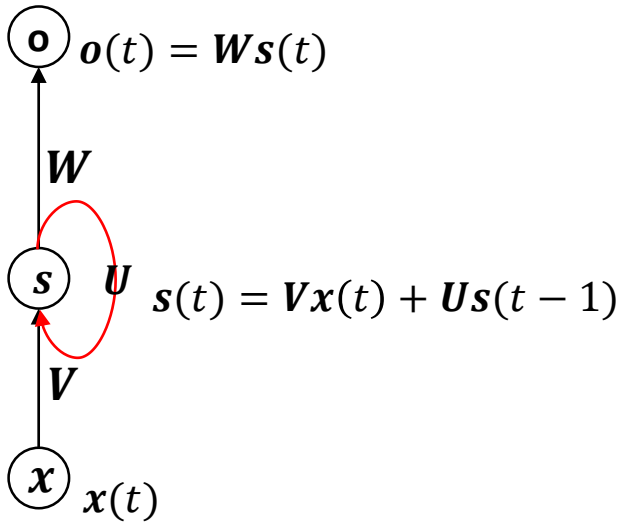




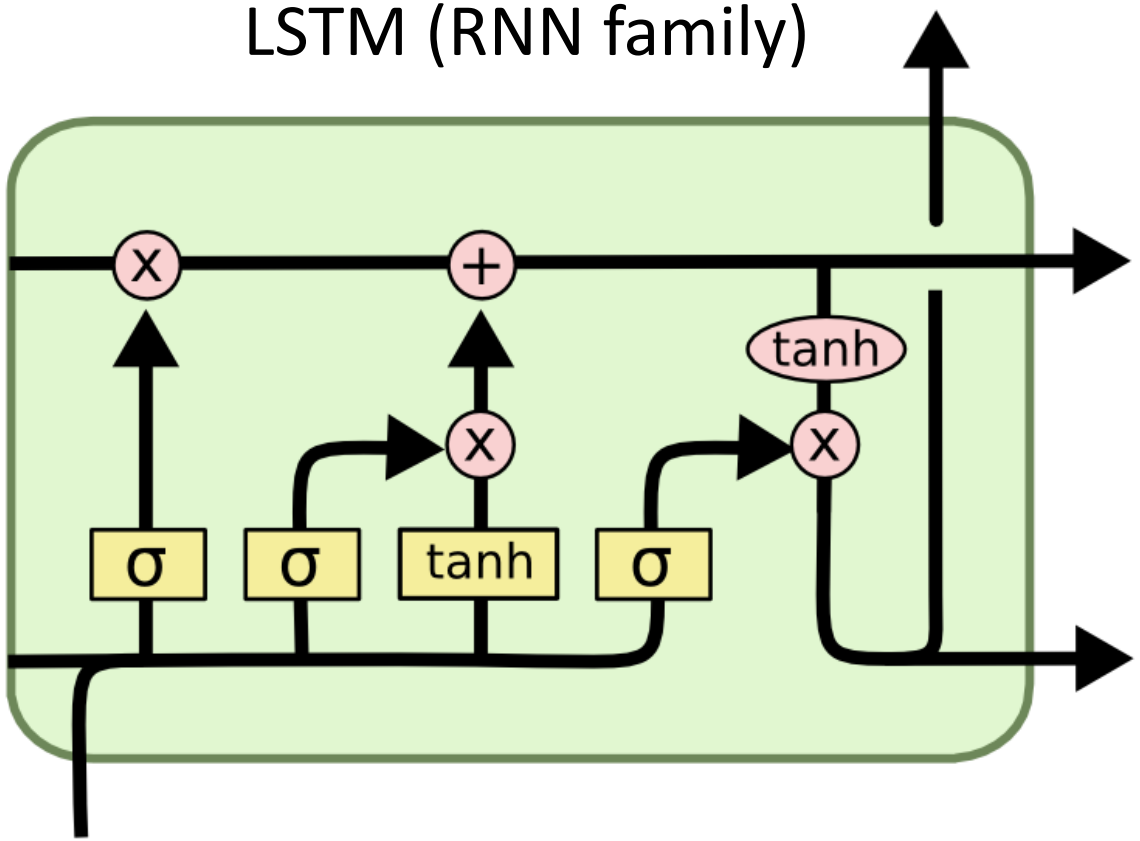
在RNN訓練期間，因為梯度會傳遞到最前一層做乘積，在前面課程提過層數一多可能會發生梯度消失和爆炸的問題，在RNN也是，傳遞不但跟層數相關和時間也先關，可以想像時間等於層數的概念，因此在RNN是很可能發生梯度消失/爆炸。

Long Short Term Memory

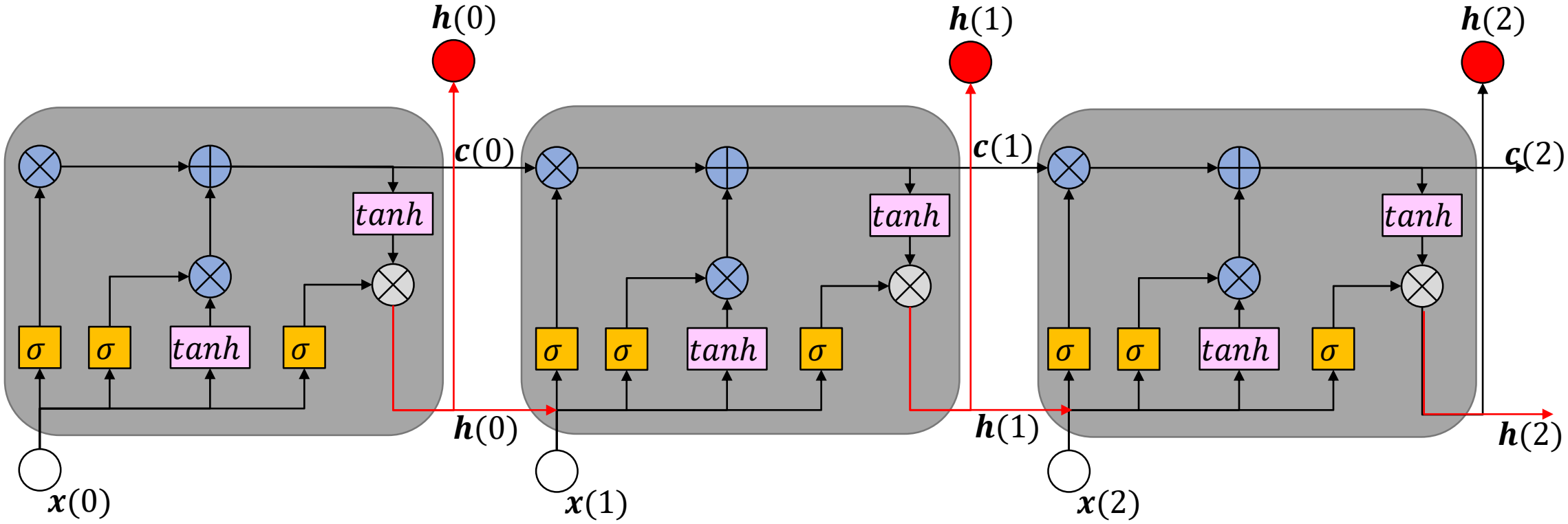
RNN



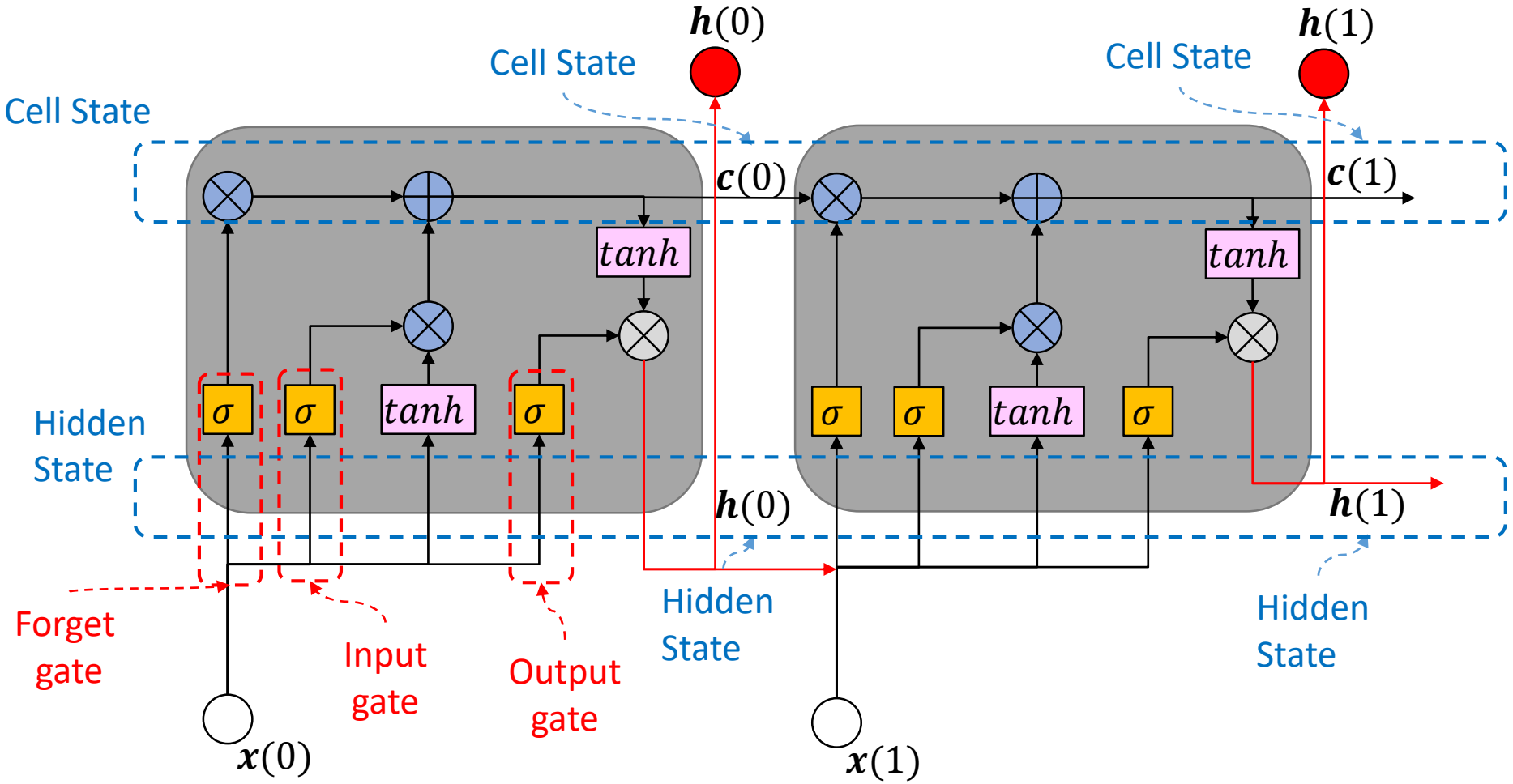
LSTM (RNN family)



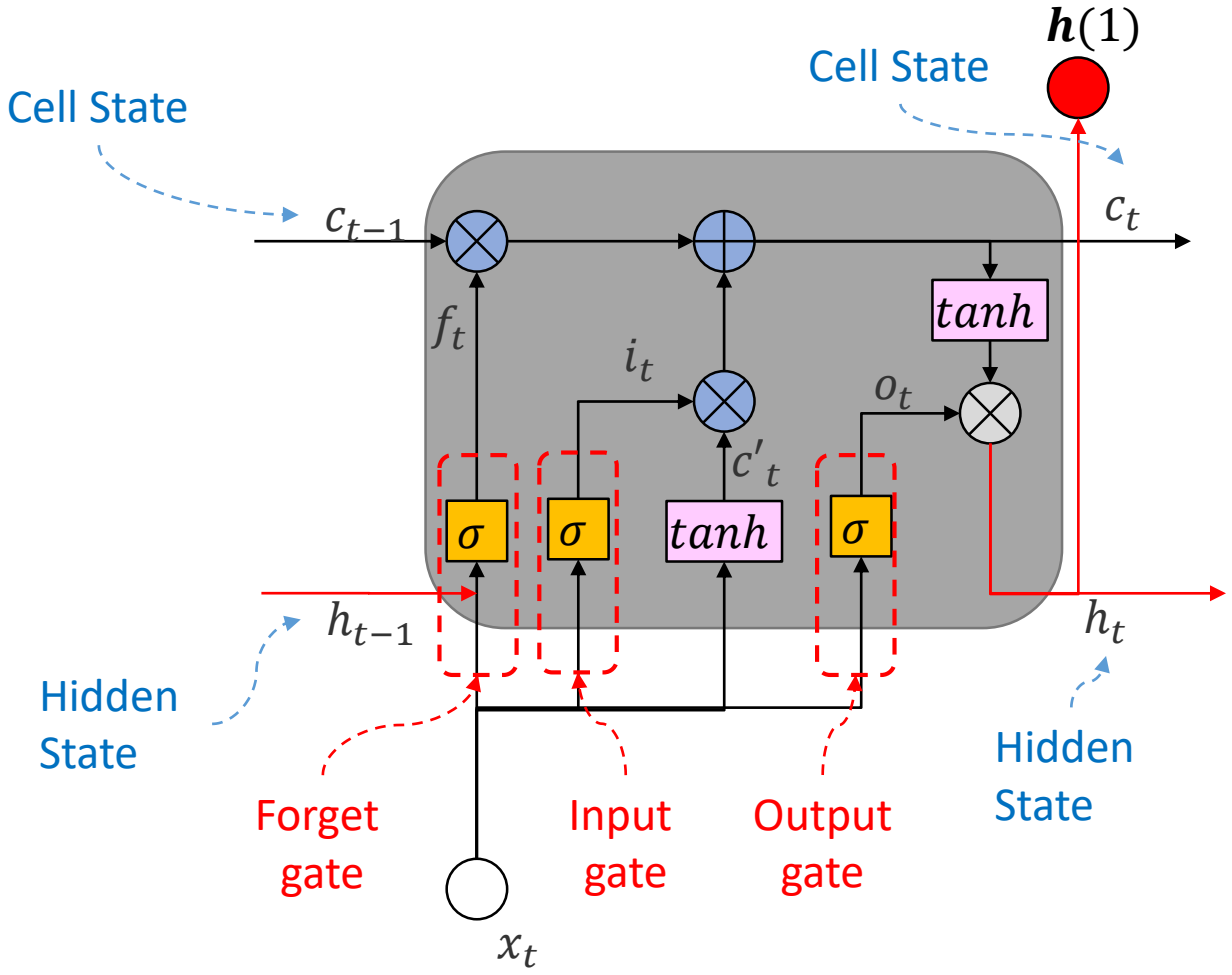
Long Short Term Memory



Long Short Term Memory



Long Short Term Memory



forget gate

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right)$$

Input gate

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$

Cell State

$$c'_t = tanh(W_c x_t + U_c h_{t-1})$$

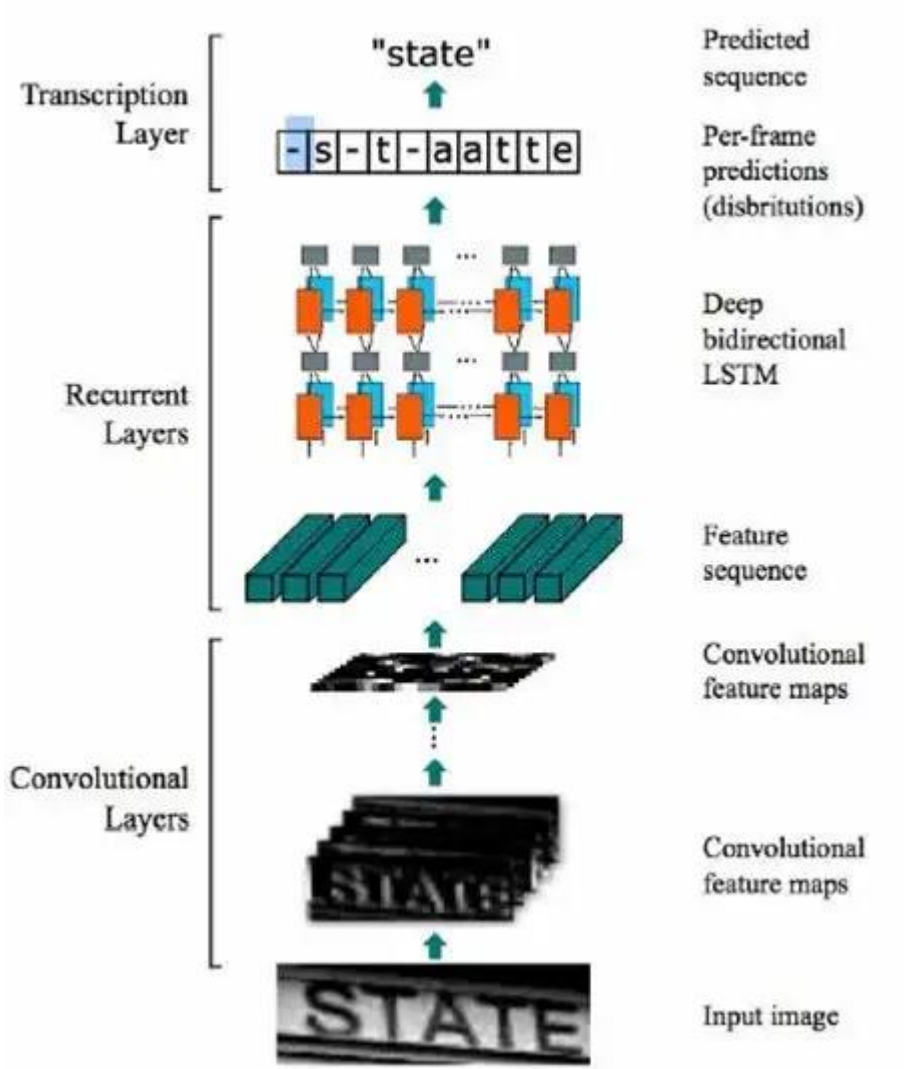
$$c_t = (f_t c_{t-1} + i_t c'_t)$$

Output gate

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$

$$h_t = o_t \tanh(c_t)$$

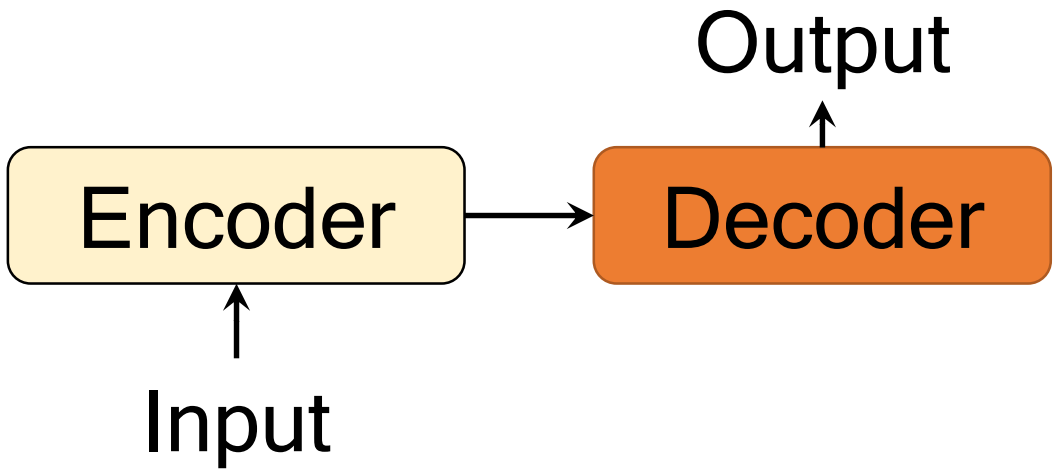
Convolutional Recurrent Neural Network)

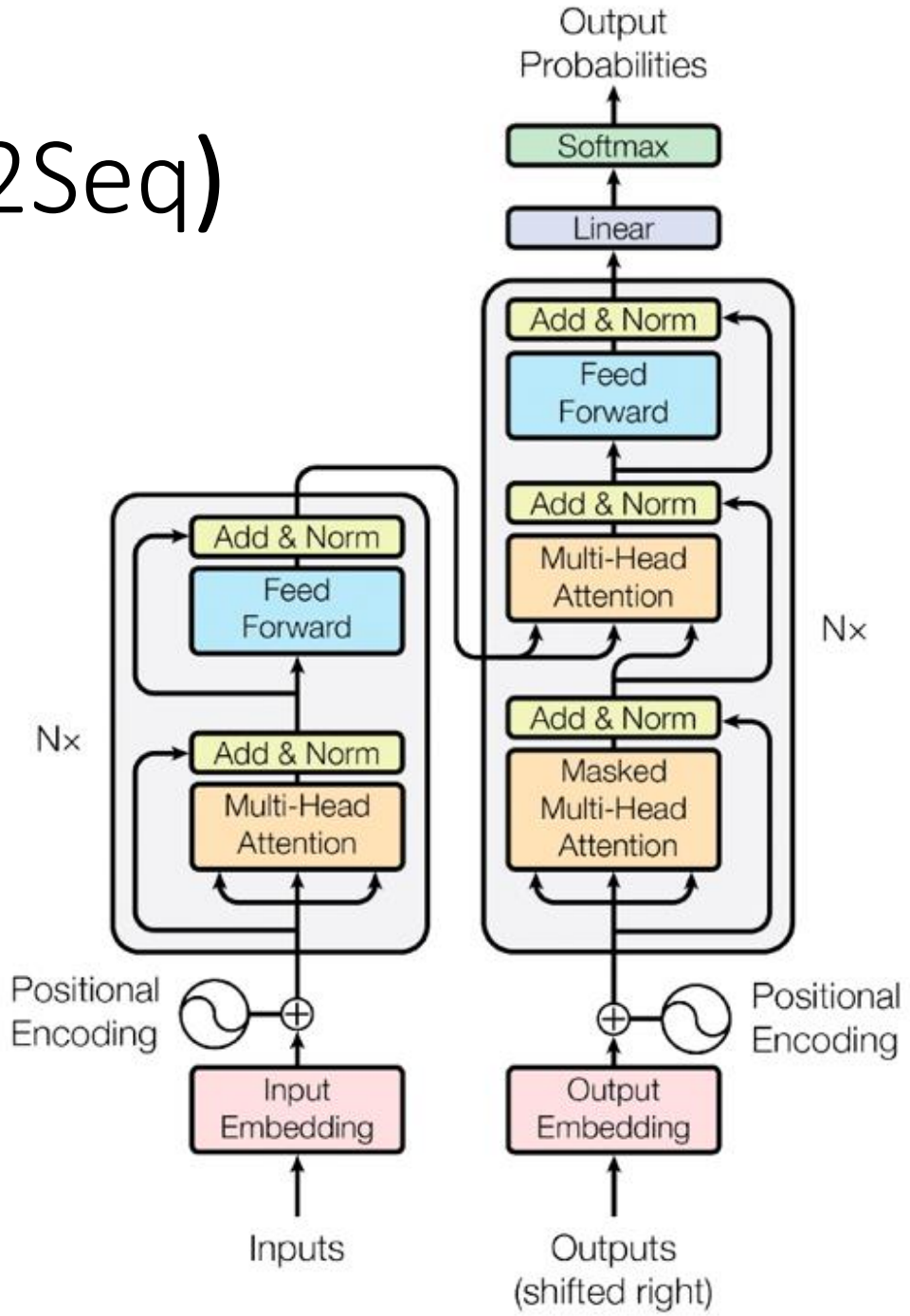


來源: https://github.com/qjadud1994/CRNN-Keras

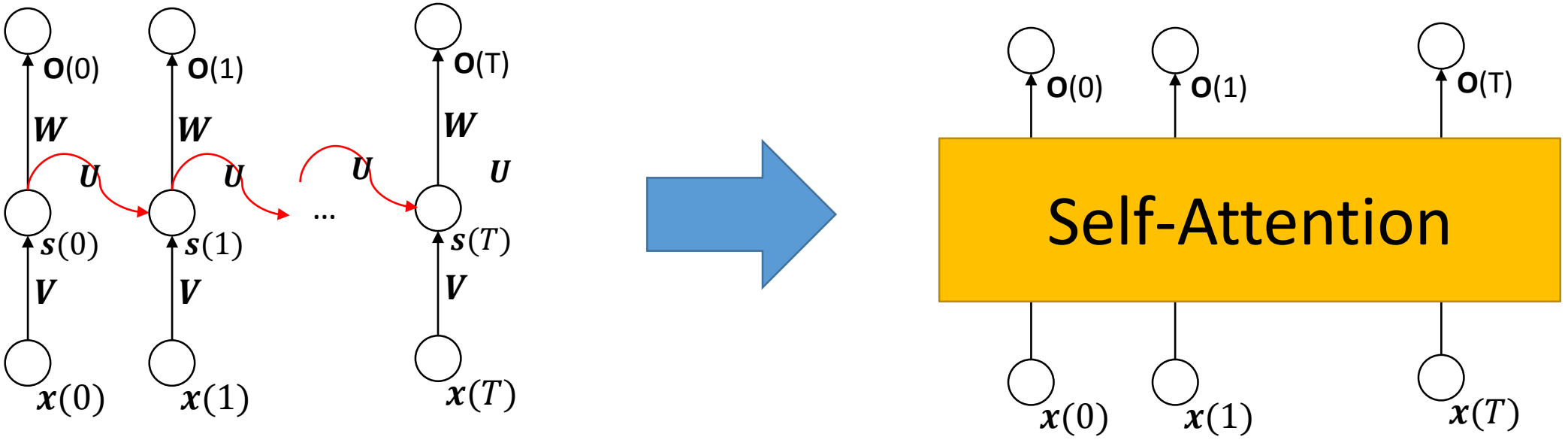
Transformer (Seq2Seq)

- Sequence-to-sequence: Seq2Seq
- Encoder: takes the input sequence and maps it into a higher dimensional space (n-dimensional vector).
- That abstract vector is fed into the Decoder which turns it into an output sequence.

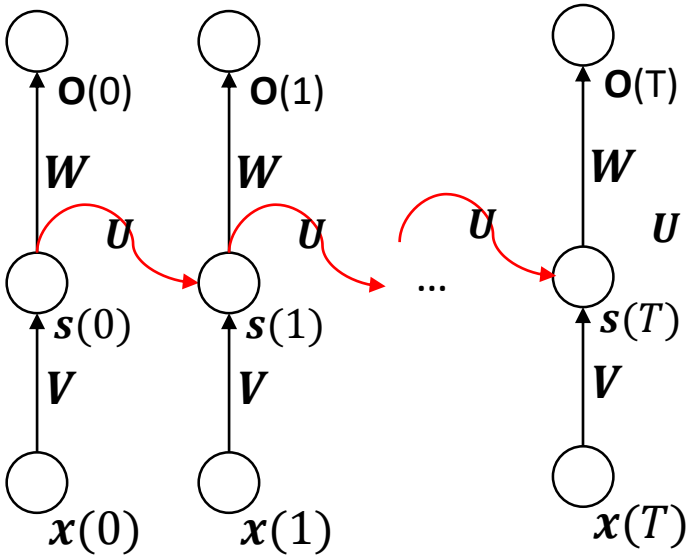


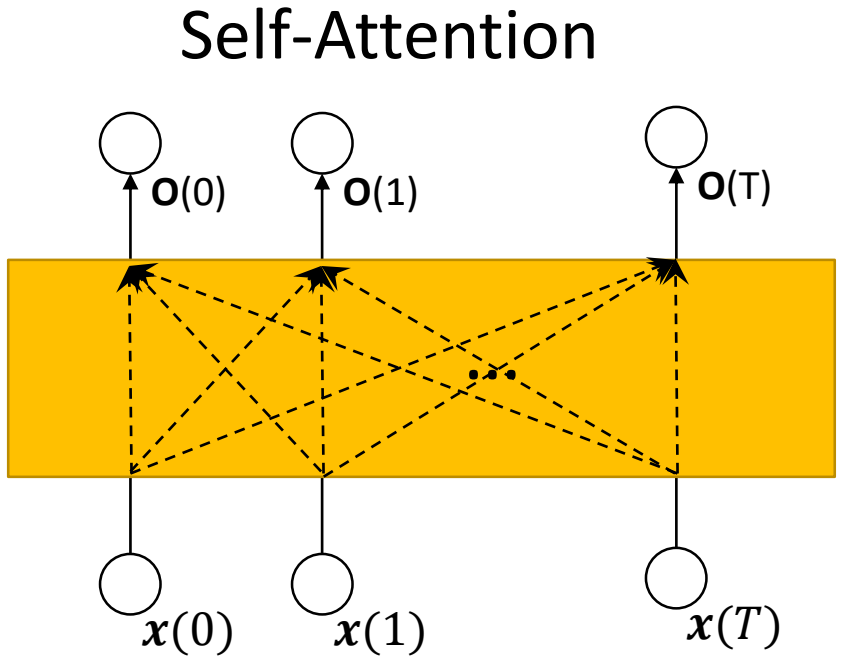


Transformer: Self-Attention



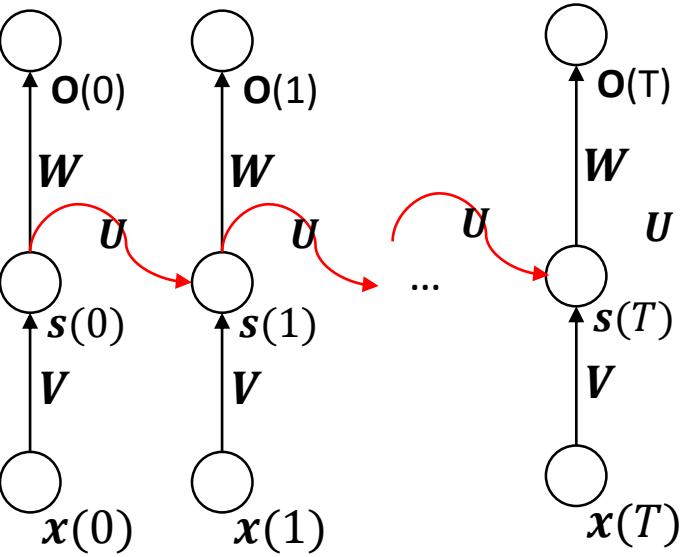
Transformer: Self-Attention

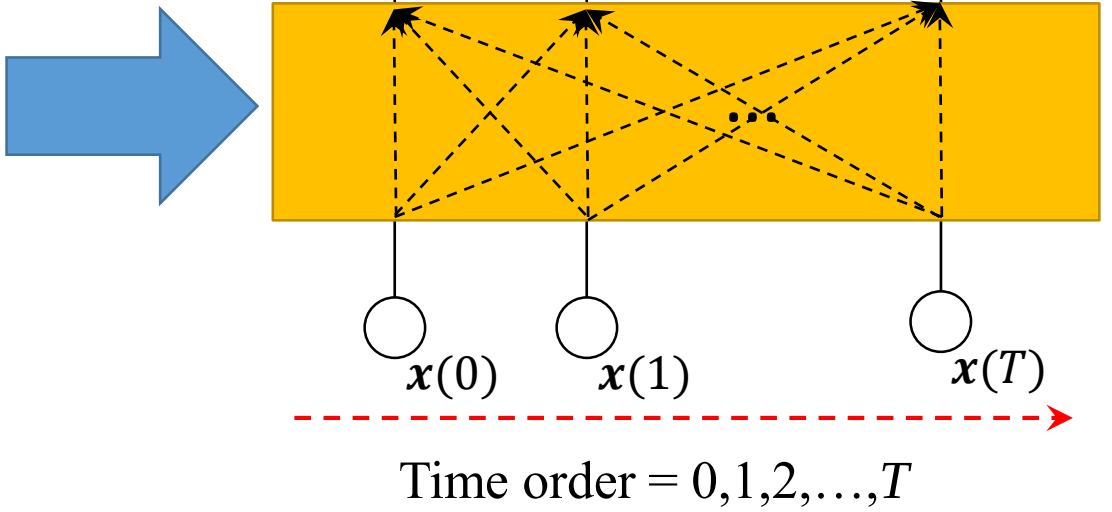


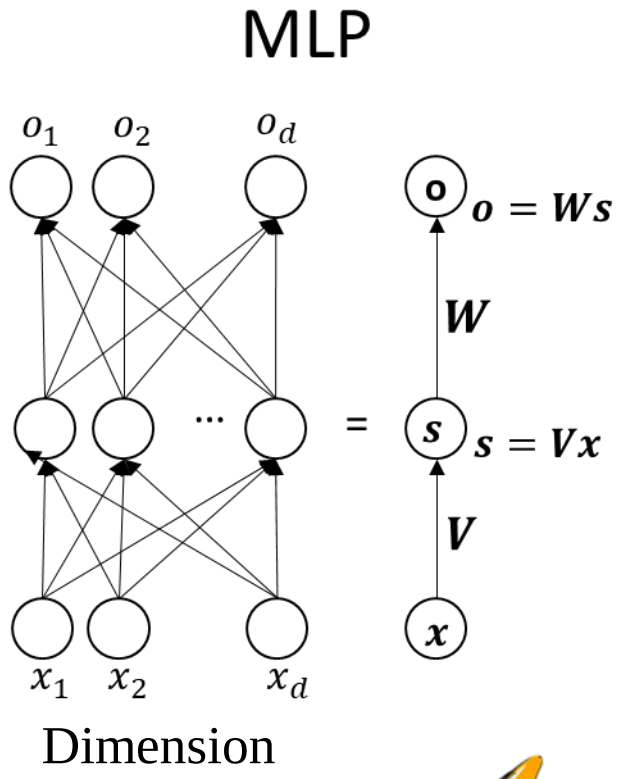


MLP?

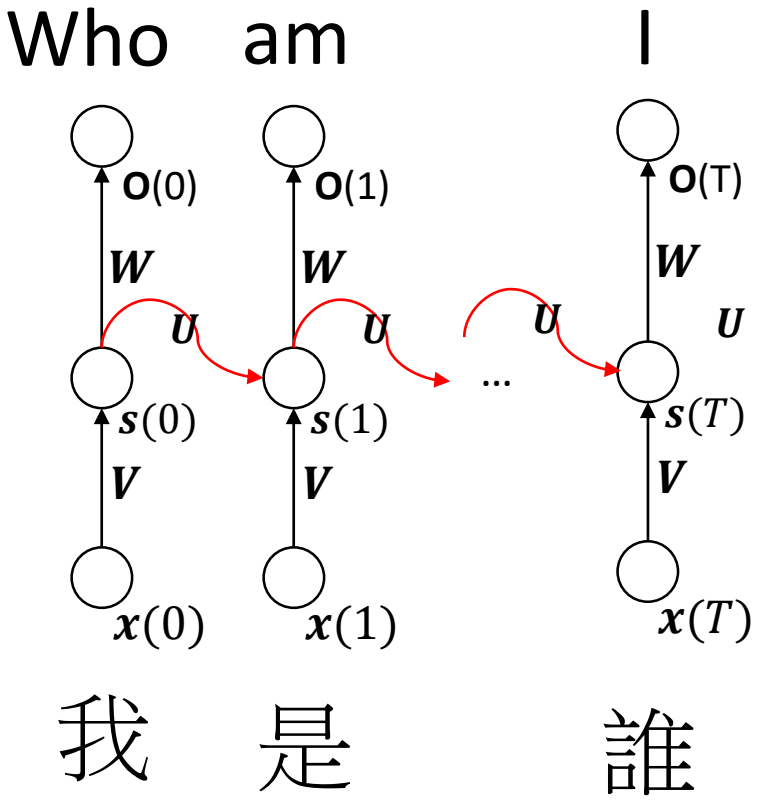
Transformer: Self-Attention

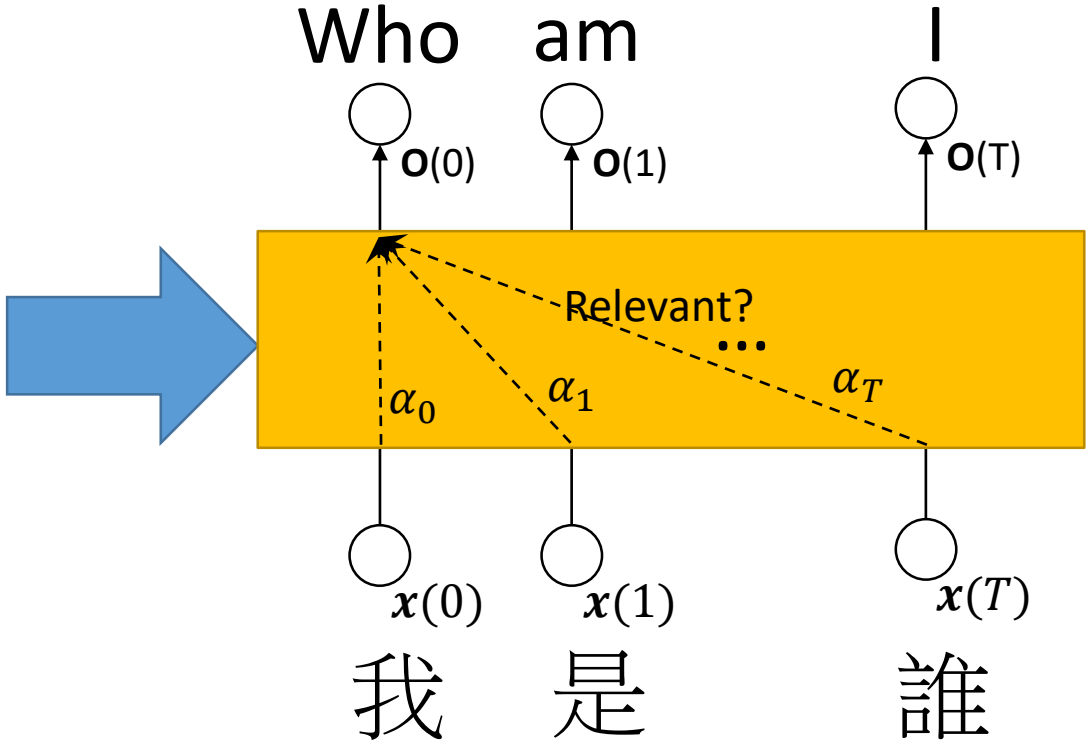


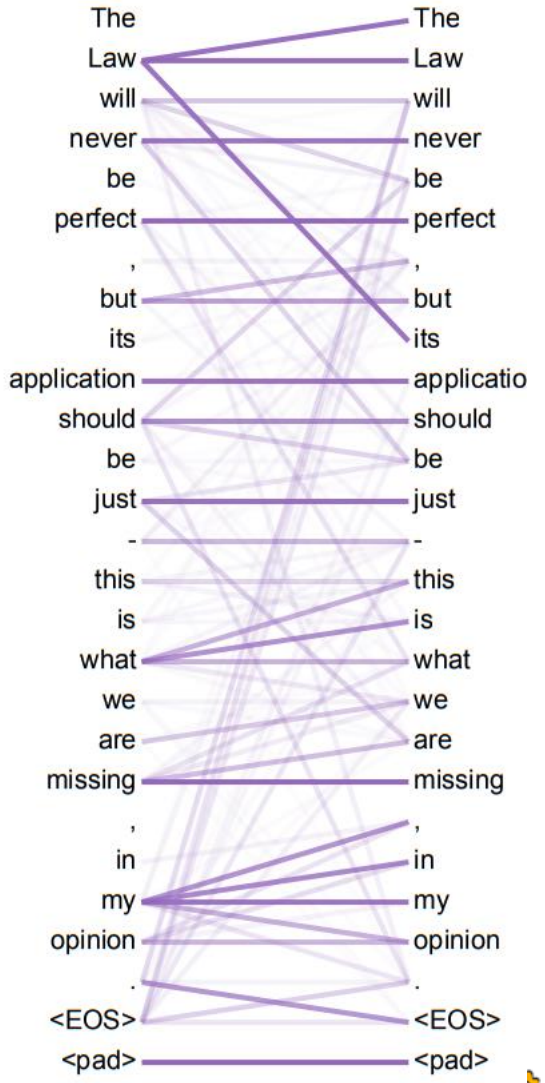




Transformer: Self-Attention

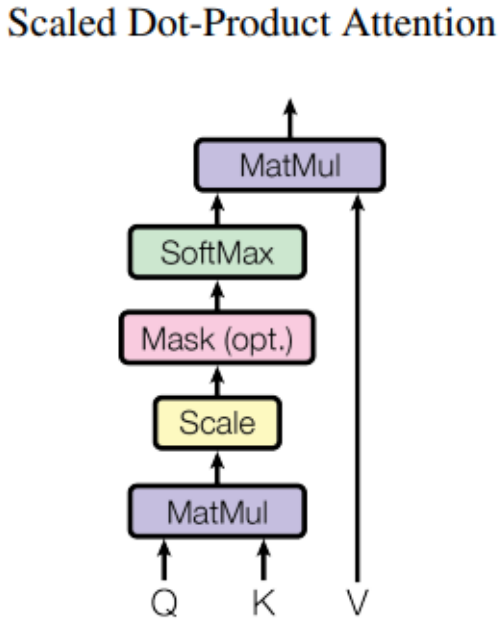




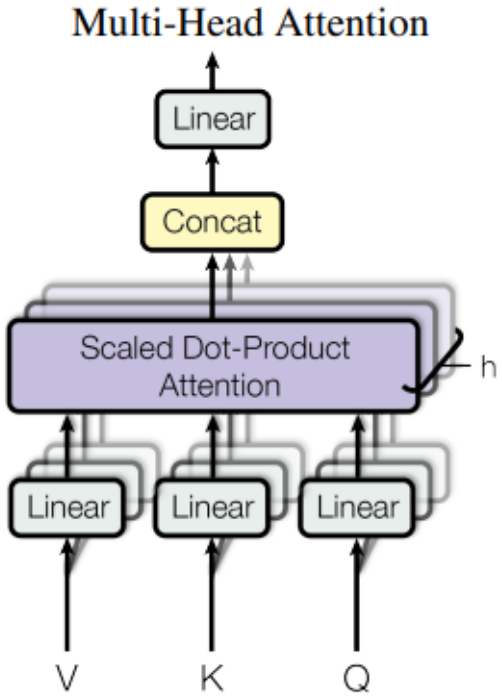


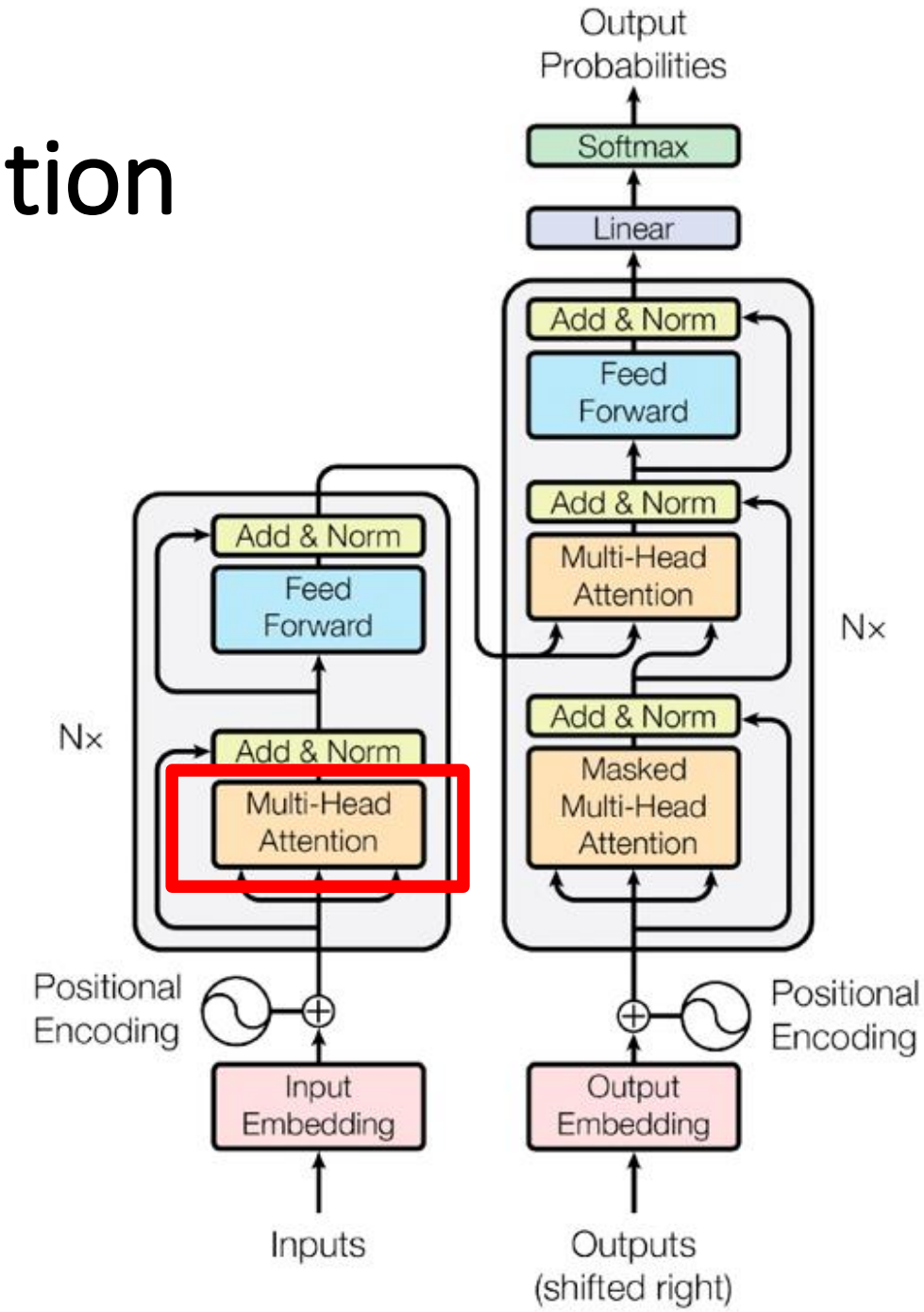
Transformer: Self-Attention

Single-Head Attention



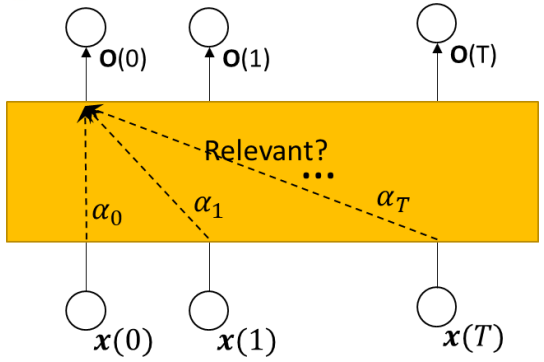
Multi-Head Attention

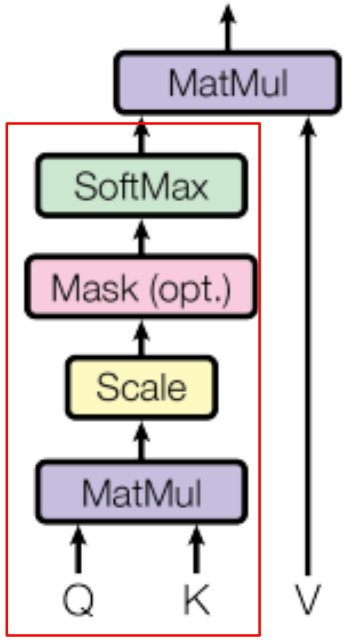




Reference: Attention Is All You Need (2017)

Single-Head Self-Attention

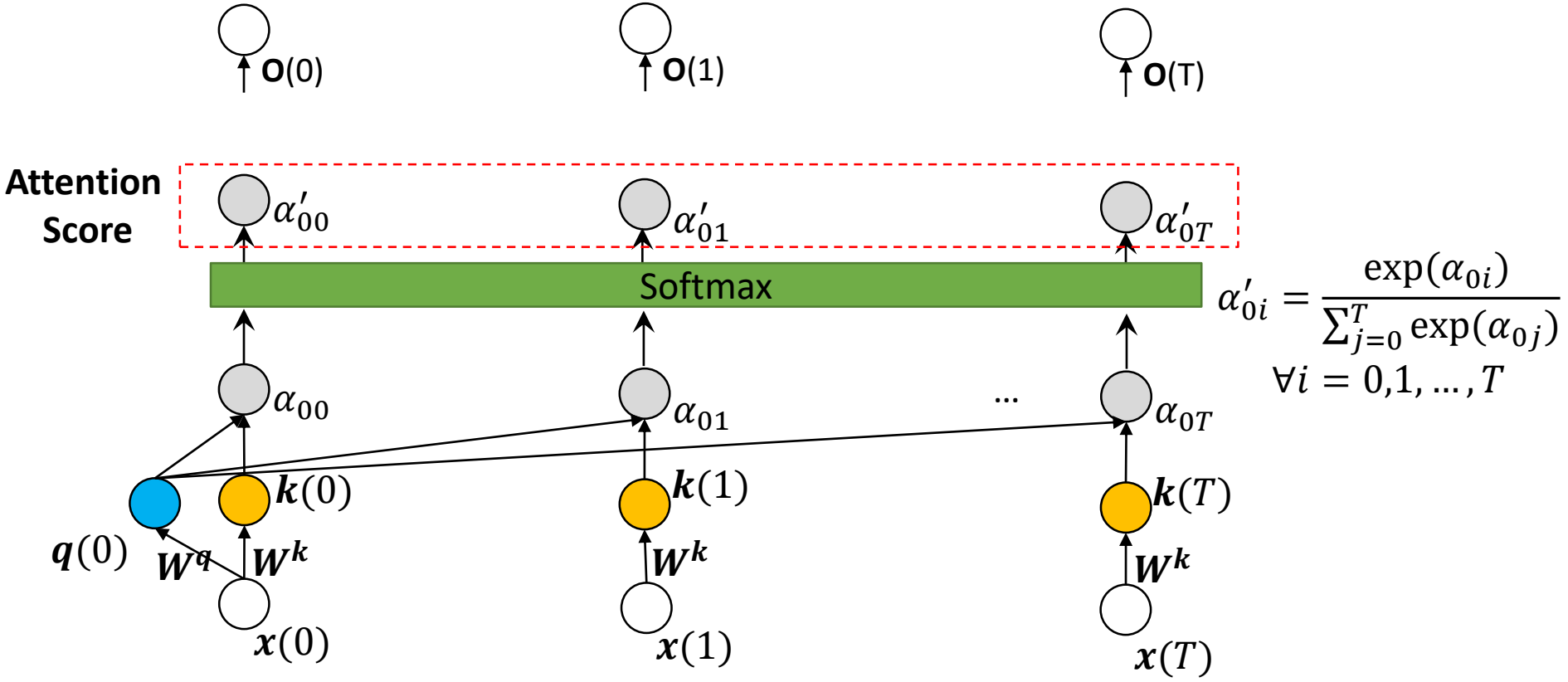




Q: Query

K: Key

V: Value



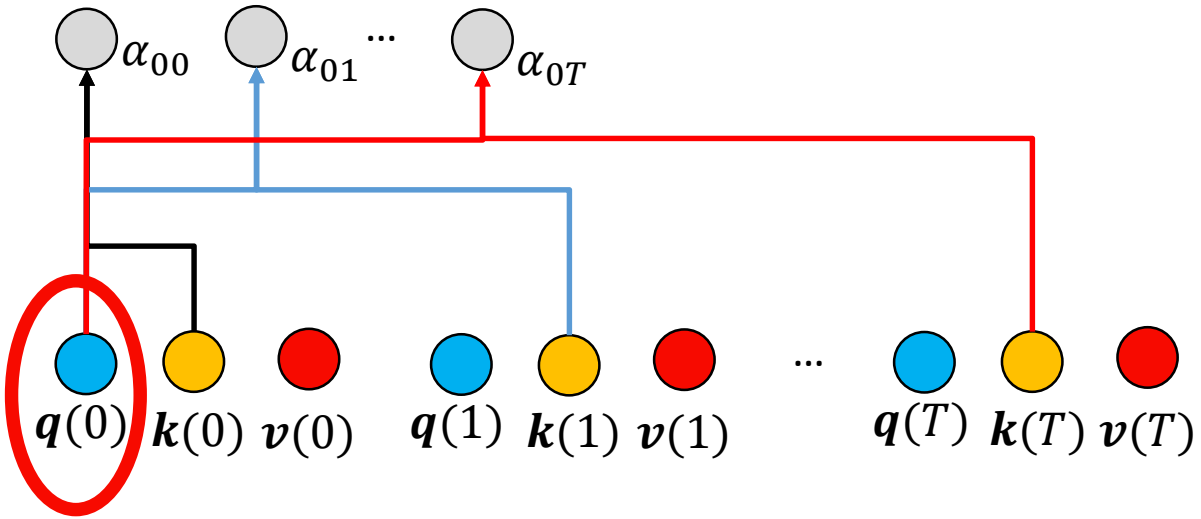
$$\alpha'_{0i} = \frac{\exp(\alpha_{0i})}{\sum_{j=0}^{T}\exp(\alpha_{0j})} \quad \forall i = 0,1,\ldots,T$$

$$q(0)=W^q x(0)$$
$$k(0)=W^k x(0)$$
$$\alpha_{00} =< q(0), k(0) >$$

$$k(1)=W^k x(1)$$
$$\alpha_{01} =< q(0), k(1) >$$

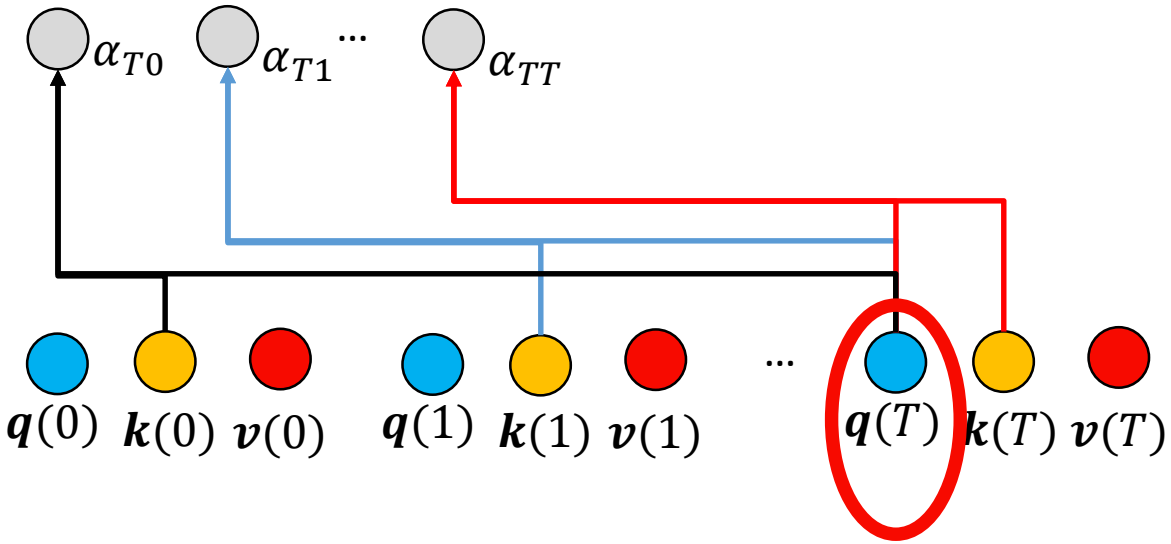
$$k(T)=W^k x(T)$$
$$\alpha_{0T} =< q(0), k(T) >$$

Attention Score



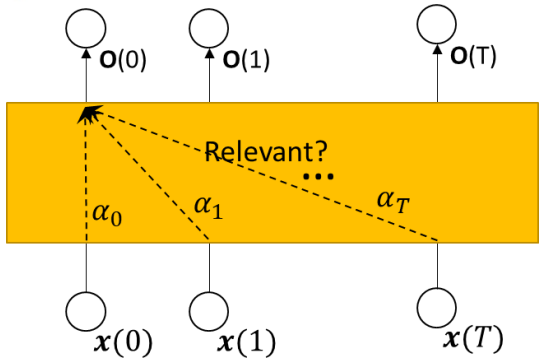
$$\alpha_{00} =< \boldsymbol{q}(0), \boldsymbol{k}(0) >$$
$$\alpha_{01} =< \boldsymbol{q}(0), \boldsymbol{k}(1) >$$
...
$$\alpha_{0T} =< \boldsymbol{q}(0), \boldsymbol{k}(T) >$$

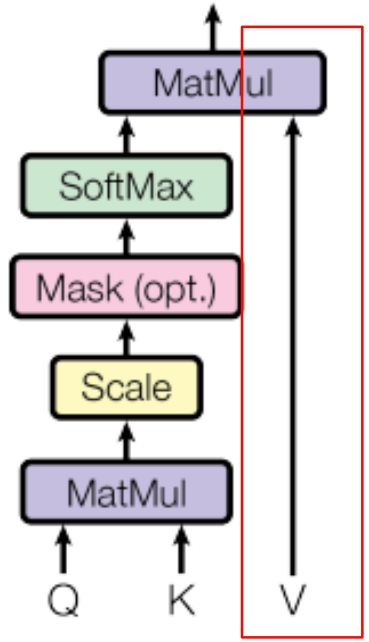
Attention Score at $q(0)$



$$\alpha_{T0} =< \boldsymbol{q}(T), \boldsymbol{k}(0) >$$
$$\alpha_{T1} =< \boldsymbol{q}(T), \boldsymbol{k}(1) >$$
...
$$\alpha_{TT} =< \boldsymbol{q}(T), \boldsymbol{k}(T) >$$

Attention Score at $q(T)$



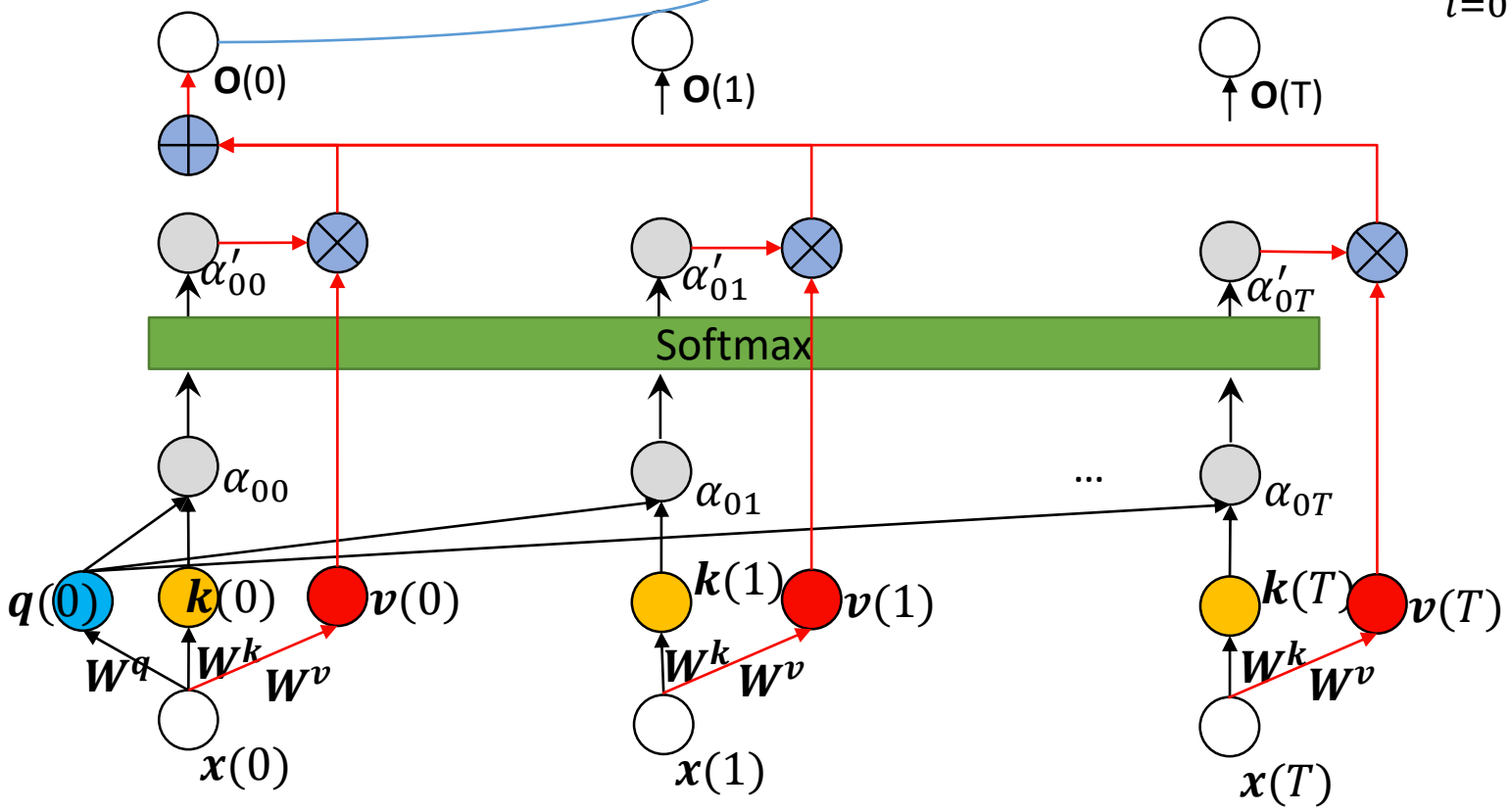


Q: Query

K: Key

V: Value

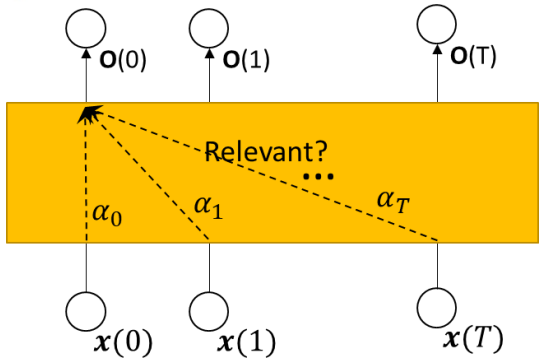
$$\boldsymbol{O}(0) = \sum_{i=0}^{T} \alpha'_{0i} v(i)$$

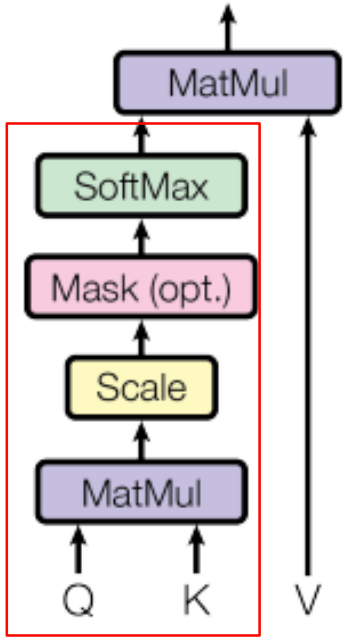


$\boldsymbol{q}(0)=\boldsymbol{W}^{\boldsymbol{q}}\boldsymbol{x}(0)$
$\boldsymbol{k}(0)=\boldsymbol{W}^{\boldsymbol{k}}\boldsymbol{x}(0)$
$\boldsymbol{v}(0)=\boldsymbol{W}^{\boldsymbol{v}}\boldsymbol{x}(0)$
$\alpha_{00} =< \boldsymbol{q}(0), \boldsymbol{k}(0) >$

$\boldsymbol{k}(1)=\boldsymbol{W}^{\boldsymbol{k}}\boldsymbol{x}(1)$
$\boldsymbol{v}(1)=\boldsymbol{W}^{\boldsymbol{v}}\boldsymbol{x}(1)$
$\alpha_{01} =< \boldsymbol{q}(0), \boldsymbol{k}(1) >$

$\boldsymbol{k}(T)=\boldsymbol{W}^{\boldsymbol{k}}\boldsymbol{x}(T)$
$\boldsymbol{v}(T)=\boldsymbol{W}^{\boldsymbol{v}}\boldsymbol{x}(T)$
$\alpha_{0T} =< \boldsymbol{q}(0), \boldsymbol{k}(T) >$



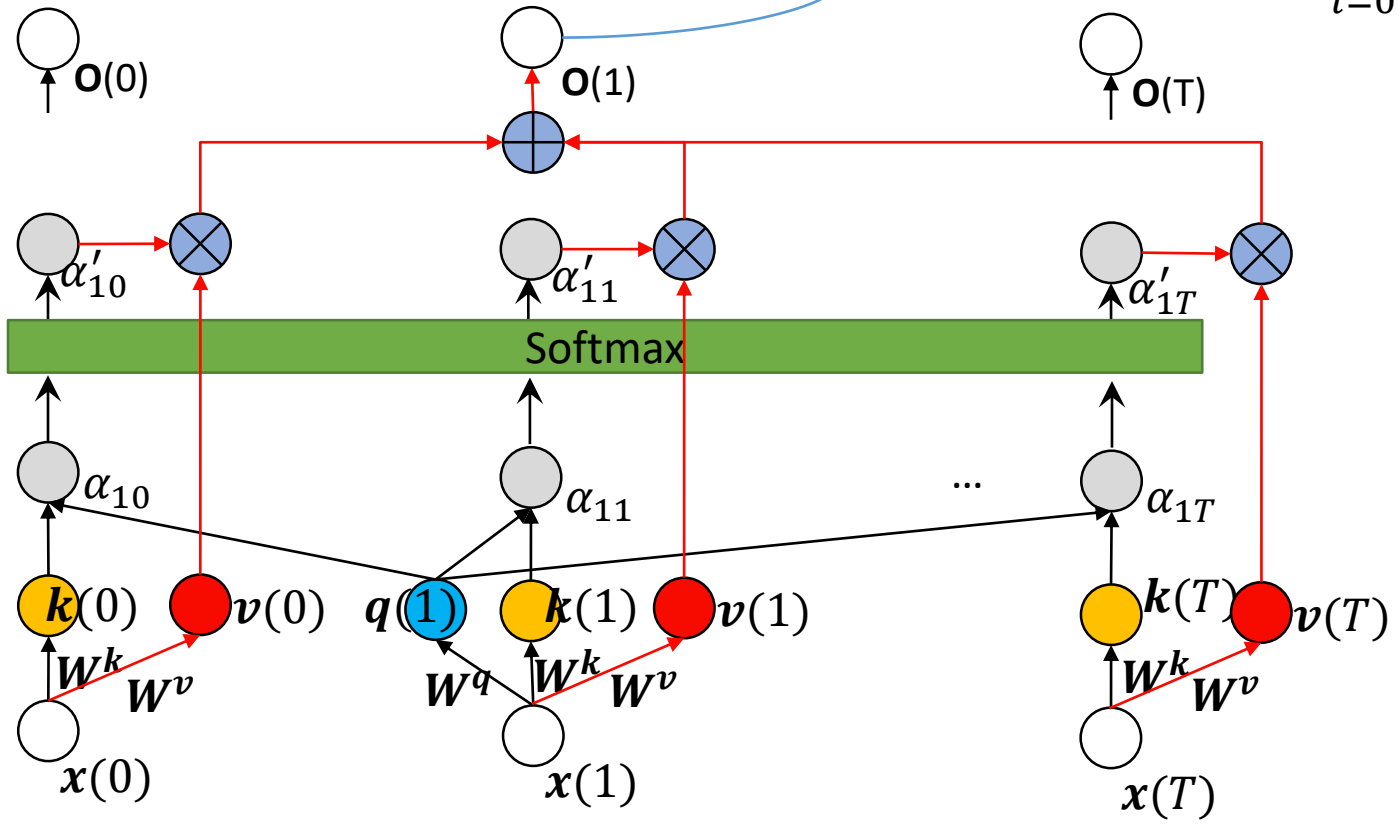


Q: Query

K: Key

V: Value

$$\boldsymbol{O}(1) = \sum_{i=0}^{T} \alpha'_{1i} v(i)$$



$\boldsymbol{k}(0)=\boldsymbol{W}^{\boldsymbol{k}}\boldsymbol{x}(0)$

$\boldsymbol{v}(0)=\boldsymbol{W}^{\boldsymbol{v}}\boldsymbol{x}(0)$

$\alpha_{10} =< \boldsymbol{q}(1), \boldsymbol{k}(0) >$

$\boldsymbol{q}(1)=\boldsymbol{W}^{\boldsymbol{q}}\boldsymbol{x}(1)$

$\boldsymbol{k}(1)=\boldsymbol{W}^{\boldsymbol{k}}\boldsymbol{x}(1)$

$\boldsymbol{v}(1)=\boldsymbol{W}^{\boldsymbol{v}}\boldsymbol{x}(1)$

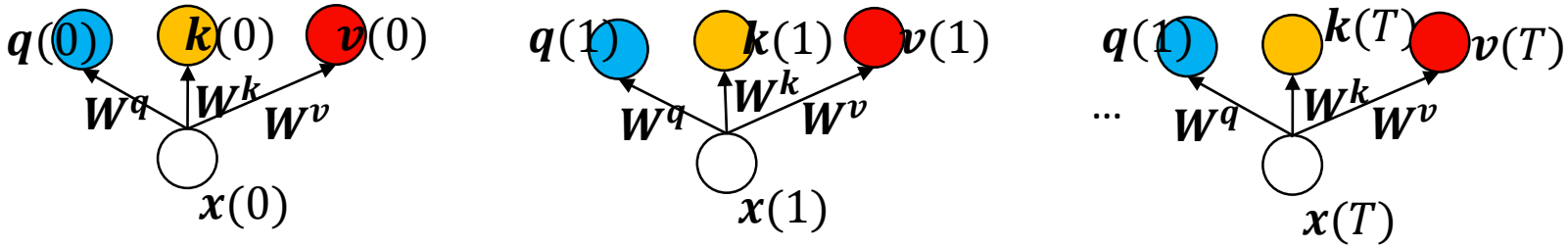
$\alpha_{11} =< \boldsymbol{q}(1), \boldsymbol{k}(1) >$

$\boldsymbol{k}(T)=\boldsymbol{W}^{\boldsymbol{k}}\boldsymbol{x}(T)$

$\boldsymbol{v}(T)=\boldsymbol{W}^{\boldsymbol{v}}\boldsymbol{x}(T)$

$\alpha_{1T} =< \boldsymbol{q}(1), \boldsymbol{k}(T) >$

用大矩陣直接運算QKV



$$q(0)=W^q x(0)$$
$$k(0)=W^k x(0)$$
$$v(0)=W^v x(0)$$

$$q(1)=W^q x(1)$$
$$k(1)=W^k x(1)$$
$$v(1)=W^v x(1)$$

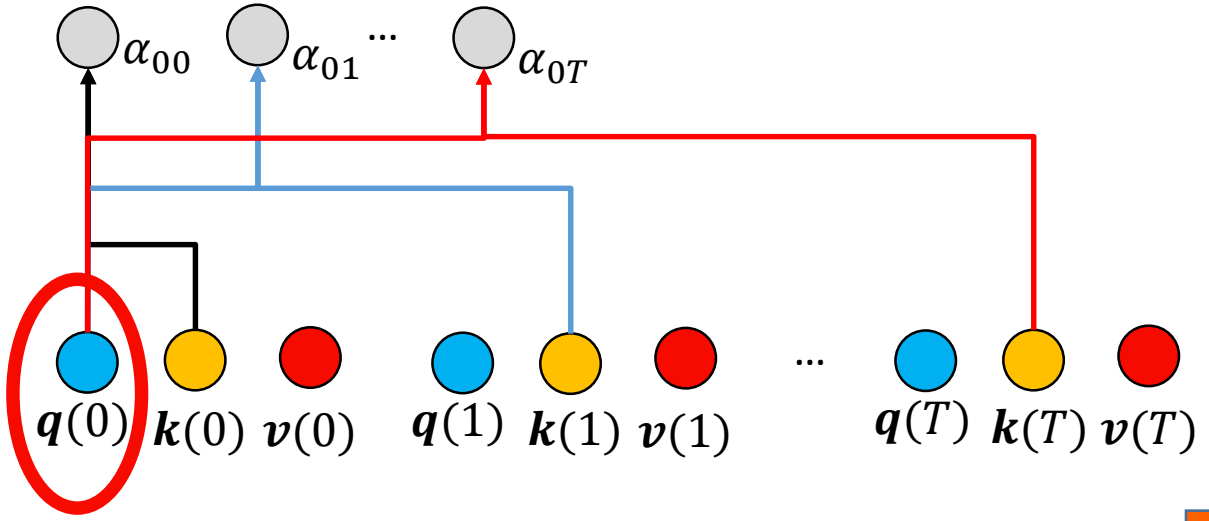
$$q(T)=W^q x(T)$$
$$k(T)=W^k x(T)$$
$$v(T)=W^v x(T)$$

$$\underset{d\times T}{Q} = \underset{d\times T}{\left[\underset{d\times 1}{q(0)}\ \underset{d\times 1}{q(1)}\ \cdots\ \underset{d\times 1}{q(T)}\right]} = \underset{d\times d}{W^q}\ \underset{d\times T}{\left[\underset{d\times 1}{x(0)}\ \underset{d\times 1}{x(1)}\ \cdots\ \underset{d\times 1}{x(T)}\right]}$$

$$\underset{d\times T}{K} = \underset{d\times T}{\left[\underset{d\times 1}{k(0)}\ \underset{d\times 1}{k(1)}\ \cdots\ \underset{d\times 1}{k(T)}\right]} = \underset{d\times d}{W^k}\ \underset{d\times T}{\left[\underset{d\times 1}{x(0)}\ \underset{d\times 1}{x(1)}\ \cdots\ \underset{d\times 1}{x(T)}\right]}$$

$$\underset{d\times T}{V} = \underset{d\times T}{\left[\underset{d\times 1}{v(0)}\ \underset{d\times 1}{v(1)}\ \cdots\ \underset{d\times 1}{v(T)}\right]} = \underset{d\times d}{W^v}\ \underset{d\times T}{\left[\underset{d\times 1}{x(0)}\ \underset{d\times 1}{x(1)}\ \cdots\ \underset{d\times 1}{x(T)}\right]}$$

用大矩陣直接運算A



$$\alpha_{00} =< \boldsymbol{q}(0), \boldsymbol{k}(0) >$$
$$\alpha_{01} =< \boldsymbol{q}(0), \boldsymbol{k}(1) >$$
$$\dots$$
$$\alpha_{0T} =< \boldsymbol{q}(0), \boldsymbol{k}(T) >$$

Attention Score at $q(0)$

$$\begin{bmatrix} \alpha_{00} & \alpha_{10} & \cdots & \alpha_{T0} \\ \alpha_{01} & \alpha_{11} & \cdots & \alpha_{T1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{0T} & \alpha_{1T} & \cdots & \alpha_{TT} \end{bmatrix}$$

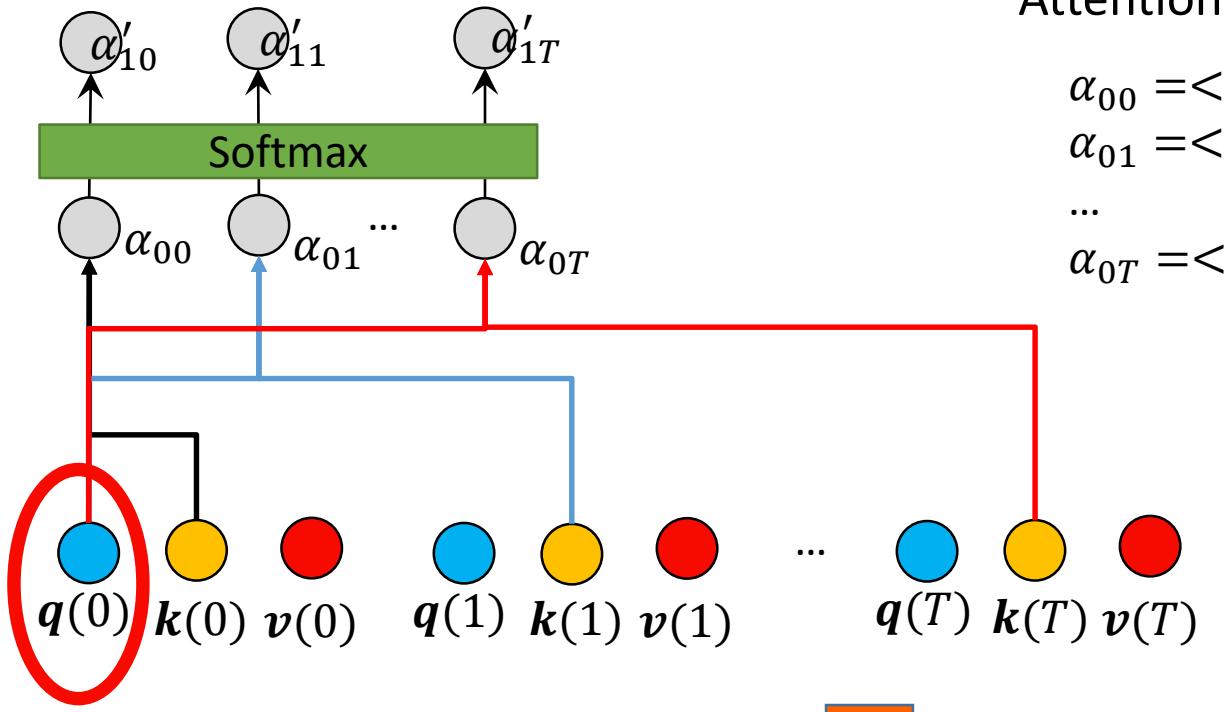
Attention Score at $q(0)$

Attention Score Matrix

$$= \underset{T\times T}{\boldsymbol{A}} =< \boldsymbol{Q}, \boldsymbol{K} > = \underset{T\times d}{\boldsymbol{K}^{\boldsymbol{T}}} \underset{d\times T}{\boldsymbol{Q}} =$$

$$\begin{bmatrix} \boldsymbol{k}(0)_{d\times 1} \\ \boldsymbol{k}(1)_{d\times 1} \\ \boldsymbol{k}(T)_{d\times 1} \end{bmatrix}_{T\times d} \begin{bmatrix} \underset{d\times 1}{\boldsymbol{q}(0)} & \underset{d\times 1}{\boldsymbol{q}(1)} & \cdots & \underset{d\times 1}{\boldsymbol{q}(T)} \end{bmatrix}_{d\times T}$$

$\mathbf{K}^T$ $T\times d$ $\quad$ $\mathbf{Q}$ $d\times T$



Attention Score at $q(0)$

$$\alpha_{00} =< \boldsymbol{q}(0), \boldsymbol{k}(0) >$$
$$\alpha_{01} =< \boldsymbol{q}(0), \boldsymbol{k}(1) >$$
$$\dots$$
$$\alpha_{0T} =< \boldsymbol{q}(0), \boldsymbol{k}(T) >$$

Softmax

$$\alpha'_{0i} = \frac{\exp(\alpha_{0i})}{\sum_{j=0}^{T} \exp(\alpha_{0j})} \quad \forall i = 0,1,\dots,T$$

Attention Score Matrix

$$\begin{bmatrix} \alpha_{00} & \alpha_{10} & \cdots & \alpha_{T0} \\ \alpha_{01} & \alpha_{11} & \cdots & \alpha_{T1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{0T} & \alpha_{1T} & \cdots & \alpha_{TT} \end{bmatrix} = \underset{T \times T}{\boldsymbol{A}}$$

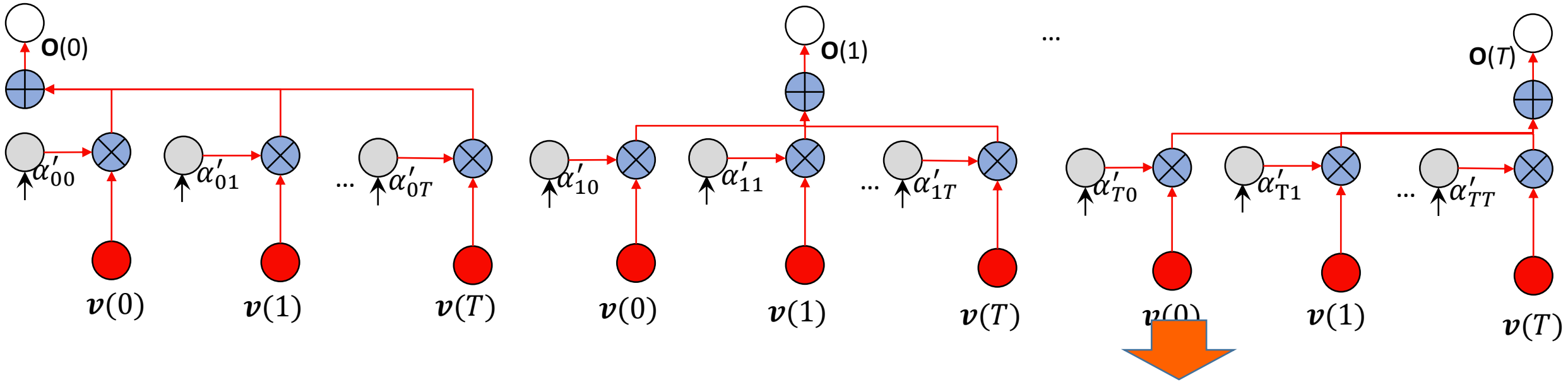
Softmax
For each column

$$\underset{T \times T}{\hat{\boldsymbol{A}}} = \begin{bmatrix} \alpha'_{00} & \alpha'_{10} & \cdots & \alpha'_{T0} \\ \alpha'_{01} & \alpha'_{11} & \cdots & \alpha'_{T1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha'_{0T} & \alpha'_{1T} & \cdots & \alpha'_{TT} \end{bmatrix}$$

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

Attention Score at $q(0)$

用大矩陣直接運算O



$$\underset{d\times T}{\boldsymbol{O}} = \underset{d\times T}{\left[\underset{d\times 1}{\boldsymbol{o}(0)}\ \underset{d\times 1}{\boldsymbol{o}(1)}\ \cdots\ \underset{d\times 1}{\boldsymbol{o}(T)}\right]} = \underset{d\times T\ \ T\times T}{\boldsymbol{V}\widehat{\boldsymbol{A}}} = \underset{d\times T}{\left[\underset{d\times 1}{\boldsymbol{v}(0)}\ \underset{d\times 1}{\boldsymbol{v}(1)}\ \cdots\ \underset{d\times 1}{\boldsymbol{v}(T)}\right]} \underset{T\times T}{\begin{bmatrix} \alpha'_{00} & \alpha'_{10} & \cdots & \alpha'_{T0} \\ \alpha'_{01} & \alpha'_{11} & \cdots & \alpha'_{T1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha'_{0T} & \alpha'_{1T} & \cdots & \alpha'_{TT} \end{bmatrix}}$$

矩陣直接運算Self-Attention

$$\underset{d\times T}{\mathbf{Q}} = \underset{d\times T}{\left[\underset{d\times 1}{\boldsymbol{q}(0)}\ \underset{d\times 1}{\boldsymbol{q}(1)}\ \cdots\ \underset{d\times 1}{\boldsymbol{q}(T)}\right]} = \underset{d\times d}{\boldsymbol{W^q}} \underset{d\times T}{\left[\underset{d\times 1}{\boldsymbol{x}(0)}\ \underset{d\times 1}{\boldsymbol{x}(1)}\ \cdots\ \underset{d\times 1}{\boldsymbol{x}(T)}\right]}$$

$$\underset{d\times T}{\mathbf{K}} = \underset{d\times T}{\left[\underset{d\times 1}{\boldsymbol{k}(0)}\ \underset{d\times 1}{\boldsymbol{k}(1)}\ \cdots\ \underset{d\times 1}{\boldsymbol{k}(T)}\right]} = \underset{d\times d}{\boldsymbol{W^k}} \underset{d\times T}{\left[\underset{d\times 1}{\boldsymbol{x}(0)}\ \underset{d\times 1}{\boldsymbol{x}(1)}\ \cdots\ \underset{d\times 1}{\boldsymbol{x}(T)}\right]}$$

$$\underset{d\times T}{\mathbf{V}} = \underset{d\times T}{\left[\underset{d\times 1}{\boldsymbol{v}(0)}\ \underset{d\times 1}{\boldsymbol{v}(1)}\ \cdots\ \underset{d\times 1}{\boldsymbol{v}(T)}\right]} = \underset{d\times d}{\boldsymbol{W^v}} \underset{d\times T}{\left[\underset{d\times 1}{\boldsymbol{x}(0)}\ \underset{d\times 1}{\boldsymbol{x}(1)}\ \cdots\ \underset{d\times 1}{\boldsymbol{x}(T)}\right]}$$

$$\underset{T\times T}{\boldsymbol{A}} = <\boldsymbol{Q},\boldsymbol{K}> = \underset{T\times d}{\boldsymbol{K^T}}\ \underset{d\times T}{\boldsymbol{Q}}$$

$$\widehat{\boldsymbol{A}} = \boldsymbol{softmax}(\boldsymbol{A})$$

$$\underset{d\times T}{\boldsymbol{O}} = \boldsymbol{V}\widehat{\boldsymbol{A}}$$

$$\boldsymbol{O} = \boldsymbol{V} \times \boldsymbol{softmx}(\boldsymbol{K^T Q})$$

論文寫法

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

差異在矩陣擺法，和一個$\sqrt{d_k}$

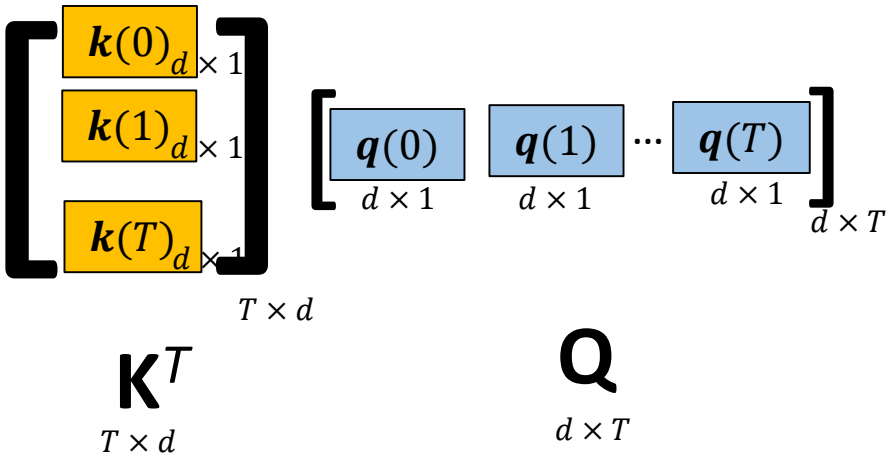
矩陣直接運算Self-Attention

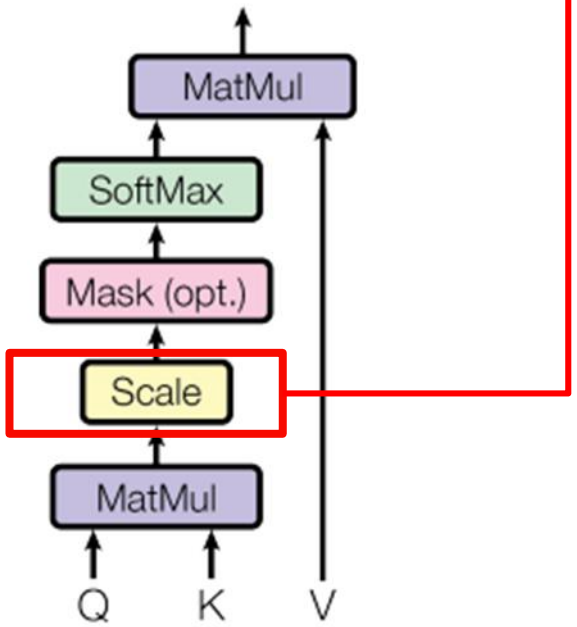
論文寫法

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

$$O = V \times softmx(K^T Q)$$

差異在矩陣擺法，和一個$\sqrt{d_k}$





K和Q在不同時間點上

d個維度相乘後的總和。

所以維度越大值越大，因此需要正規化(在右圖的Scale)，將每一個值域拉到一致。

最簡單方式有d個維度就除上d

但論文第四頁內提到當維度很大的時候，預防梯度過小的問題，因此開根號。

Self-Attention: 讓公式和論文一樣

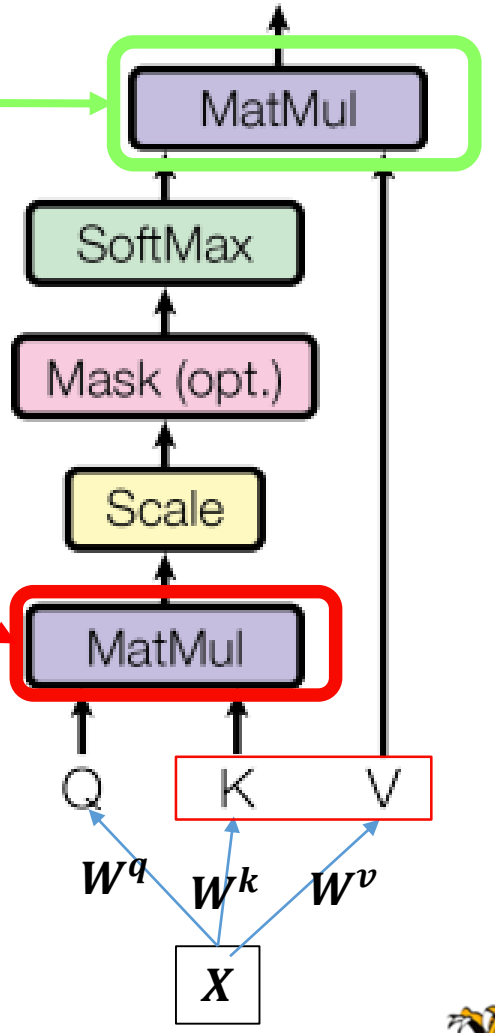
$$O = V \times softmx(K^T Q)$$

$Q_{d \times T}$ $K_{d \times T}$ $V_{d \times T}$ $O_{d \times T}$

差異在矩陣擺法

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

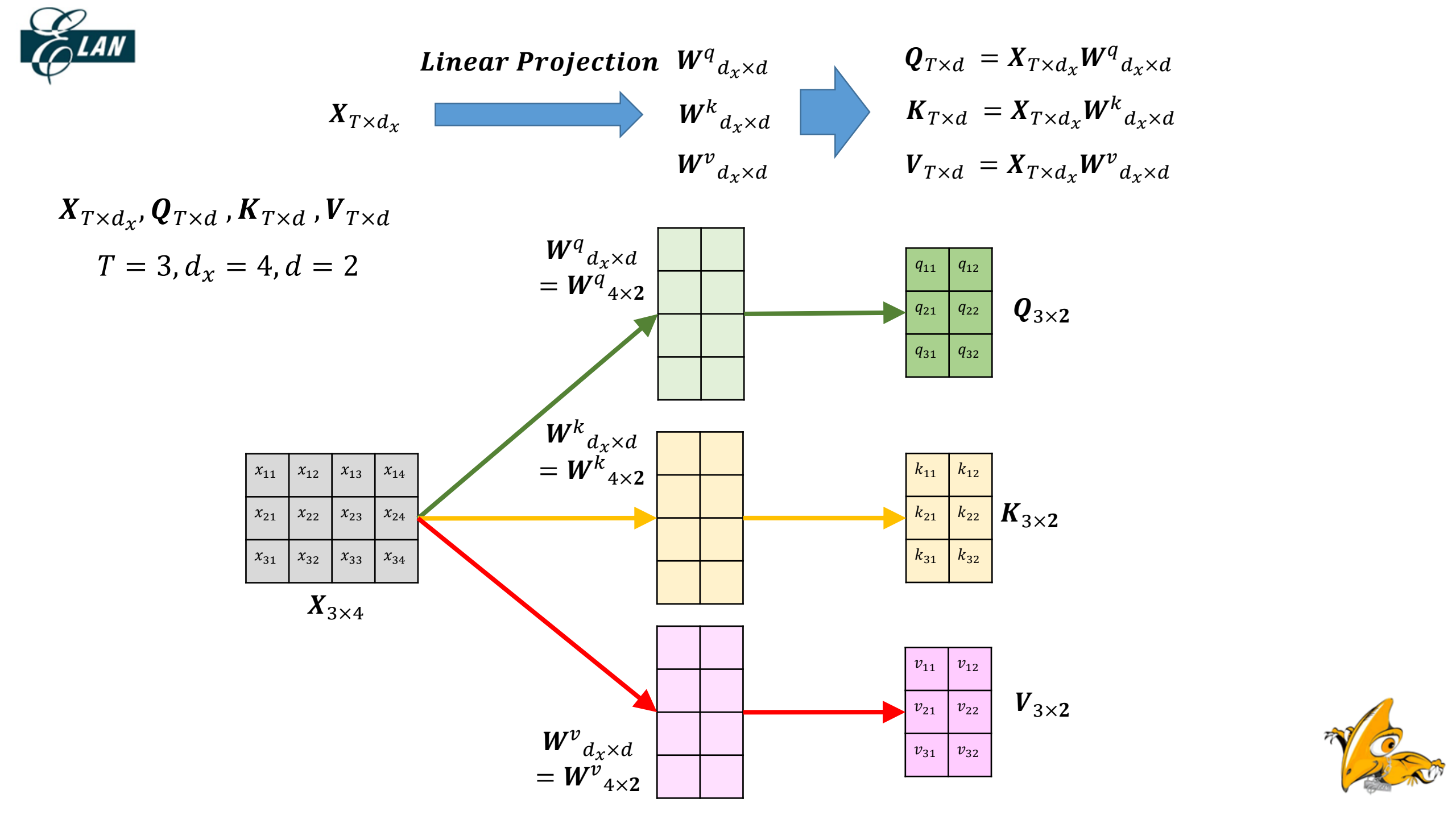
$Q_{T \times d}$ $K_{T \times d}$ $V_{T \times d}$ $O_{T \times d}$



我的擺法是讓大家比較能看得懂時序的資料是橫向的。

為了讓公式和論文一樣，我們轉成論文的公式

$$X_{T \times d_x} \xrightarrow{\textbf{Linear Projection}} \begin{matrix} W^q{}_{d_x \times d} \\ W^k{}_{d_x \times d} \\ W^v{}_{d_x \times d} \end{matrix} \Rightarrow \begin{matrix} Q_{T \times d} = X_{T \times d_x} W^q{}_{d_x \times d} \\ K_{T \times d} = X_{T \times d_x} W^k{}_{d_x \times d} \\ V_{T \times d} = X_{T \times d_x} W^v{}_{d_x \times d} \end{matrix}$$


Linear Projection
$X_{T\times d_x}$
$W^q{}_{d_x\times d}$
$W^k{}_{d_x\times d}$
$W^v{}_{d_x\times d}$
$Q_{T\times d} = X_{T\times d_x} W^q{}_{d_x\times d}$
$K_{T\times d} = X_{T\times d_x} W^k{}_{d_x\times d}$
$V_{T\times d} = X_{T\times d_x} W^v{}_{d_x\times d}$
$X_{T\times d_x}, Q_{T\times d}, K_{T\times d}, V_{T\times d}$
$T = 3, d_x = 4, d = 2$
$W^q{}_{d_x\times d} = W^q{}_{4\times 2}$
$W^k{}_{d_x\times d} = W^k{}_{4\times 2}$
$W^v{}_{d_x\times d} = W^v{}_{4\times 2}$
x_{11} x_{12} x_{13} x_{14}
x_{21} x_{22} x_{23} x_{24}
x_{31} x_{32} x_{33} x_{34}
$X_{3\times 4}$
q_{11} q_{12}
q_{21} q_{22}
q_{31} q_{32}
$Q_{3\times 2}$
k_{11} k_{12}
k_{21} k_{22}
k_{31} k_{32}
$K_{3\times 2}$
v_{11} v_{12}
v_{21} v_{22}
v_{31} v_{32}
$V_{3\times 2}$

解釋Attention Score

$$\boldsymbol{O}_{T\times d}, \boldsymbol{Q}_{T\times d}, \boldsymbol{K}_{T\times d}, \boldsymbol{V}_{T\times d}$$

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

$$\boldsymbol{O} \approx \boldsymbol{Q}\boldsymbol{K}^{\boldsymbol{T}}\boldsymbol{V}$$

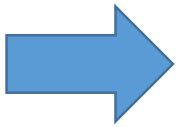
$$\boldsymbol{A}_{T_Q\times T_K} = \begin{bmatrix} a_{q_1k_1} & a_{q_1k_2} & \cdots & a_{q_1k_T} \\ a_{q_2k_1} & a_{q_2k_2} & \cdots & a_{q_2k_T} \\ \vdots & \vdots & \ddots & \vdots \\ a_{q_Tk_1} & a_{q_Tk_2} & \cdots & a_{q_Tk_T} \end{bmatrix} = \boldsymbol{Q}_{T\times d} \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1d} \\ q_{21} & q_{22} & \cdots & q_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ q_{T1} & q_{d2} & \cdots & q_{Td} \end{bmatrix} \boldsymbol{K}^{\boldsymbol{T}}{}_{d\times T} \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1T} \\ k_{21} & k_{22} & \cdots & k_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ k_{d1} & k_{d2} & \cdots & k_{dT} \end{bmatrix}$$

$a_{q_1k_1}$: Q在第1個時間資料和K第1個時間資料的內積

⋮

$a_{q_Tk_1}$: Q在第T個時間資料和K第1個時間資料的內積

$a_{q_Tk_T}$: Q在第T個時間資料和K第T個時間資料的內積



Q和K 時間資料之間的相關矩陣

解釋Attention Score

$$\boldsymbol{O}_{T\times d}, \boldsymbol{Q}_{T\times d}, \boldsymbol{K}_{T\times d}, \boldsymbol{V}_{T\times d}$$

$$T = 3, d = 2$$

$$\boldsymbol{O} = softmax(\boldsymbol{QK^T})\,\boldsymbol{V} = \hat{\boldsymbol{A}}\boldsymbol{V}$$

$\boldsymbol{O}_{3\times 2}$

o_{11}	o_{12}
o_{21}	o_{22}
o_{31}	o_{32}

$\boldsymbol{Q}_{3\times 2}$

q_{11}	q_{12}
q_{21}	q_{22}
q_{31}	q_{32}

$\boldsymbol{K}_{3\times 2}$

k_{11}	k_{12}
k_{21}	k_{22}
k_{31}	k_{32}

$\boldsymbol{V}_{3\times 2}$

v_{11}	v_{12}
v_{21}	v_{22}
v_{31}	v_{32}

$\boldsymbol{QK^T}$ = $\boldsymbol{Q}$ × $\boldsymbol{K^T}$

a_{11}	a_{12}	a_{13}
a_{21}	a_{22}	a_{23}
a_{31}	a_{32}	a_{33}

=

$t=1$	q_{11}	q_{12}
$t=2$	q_{21}	q_{22}
$t=3$	q_{31}	q_{32}

×

$t=1\ t=2\ t=3$

k_{11}	k_{21}	k_{31}
k_{12}	k_{22}	k_{32}

Q和K 時間資料之間的相關矩陣

$$\hat{\boldsymbol{A}} = softmax(\boldsymbol{QK^T})$$

a'_{11}	a'_{12}	a'_{13}
a'_{21}	a'_{22}	a'_{23}
a'_{31}	a'_{32}	a'_{33}

每個row做softmax

$$a'_{11} = \frac{a_{11}}{a_{11}+a_{12}+a_{13}}$$

$$a'_{12} = \frac{a_{12}}{a_{11}+a_{12}+a_{13}}$$

$$a'_{13} = \frac{a_{13}}{a_{11}+a_{12}+a_{13}}$$

Value和Attention Score

$$\boldsymbol{O}_{T\times d}, \boldsymbol{Q}_{T\times d}, \boldsymbol{K}_{T\times d}, \boldsymbol{V}_{T\times d}$$

$$\boldsymbol{O} \approx \boldsymbol{Q}\boldsymbol{K}^{\boldsymbol{T}}\boldsymbol{V}$$

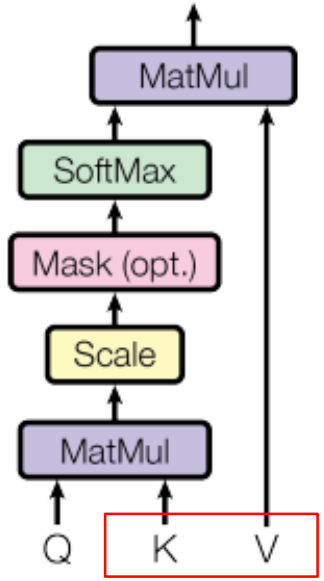
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

$$\boldsymbol{O}_{T_Q\times d} \quad \boldsymbol{A}_{T_Q\times T_K} \quad \boldsymbol{V}_{T\times d}$$

$$t = 1 \begin{bmatrix} o_{11} & o_{12} & \cdots & o_{1d} \\ o_{21} & o_{22} & \cdots & o_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ o_{T1} & o_{d2} & \cdots & o_{Td} \end{bmatrix} = \begin{bmatrix} a_{q_1k_1} & a_{q_1k_2} & \cdots & a_{q_1k_T} \\ a_{q_2k_1} & a_{q_2k_2} & \cdots & a_{q_2k_T} \\ \vdots & \vdots & \ddots & \vdots \\ a_{q_Tk_1} & a_{q_Tk_2} & \cdots & a_{q_Tk_T} \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1d} \\ v_{21} & v_{22} & \cdots & v_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ v_{T1} & v_{d2} & \cdots & v_{Td} \end{bmatrix}$$

Q第一個時間和K所有時間的係數

V第一個維度所有時間的資料



$O_{T_Q\times d}$：將V從K空間投影到Q空間

Value和Attention Score

$$\boldsymbol{O}_{T\times d}, \boldsymbol{Q}_{T\times d}, \boldsymbol{K}_{T\times d}, \boldsymbol{V}_{T\times d}$$

$$T = 3, d = 2$$

$$\boldsymbol{O} = softmax(\boldsymbol{QK^T})\,\boldsymbol{V} = \hat{\boldsymbol{A}}\boldsymbol{V}$$

$\boldsymbol{O}_{3\times 2}$

o_{11}	o_{12}
o_{21}	o_{22}
o_{31}	o_{32}

$\boldsymbol{Q}_{3\times 2}$

q_{11}	q_{12}
q_{21}	q_{22}
q_{31}	q_{32}

$\boldsymbol{K}_{3\times 2}$

k_{11}	k_{12}
k_{21}	k_{22}
k_{31}	k_{32}

$\boldsymbol{V}_{3\times 2}$

v_{11}	v_{12}
v_{21}	v_{22}
v_{31}	v_{32}

$\boldsymbol{O}_{3\times 2}$ = $\hat{\boldsymbol{A}} = softmax(\boldsymbol{QK^T})$ $\boldsymbol{V}_{3\times 2}$

o_{11}	o_{12}	=	a'_{11}	a'_{12}	a'_{13}	×	v_{11}	v_{12}
o_{21}	o_{22}		a'_{21}	a'_{22}	a'_{23}		v_{21}	v_{22}
o_{31}	o_{32}		a'_{31}	a'_{32}	a'_{33}		v_{31}	v_{32}

$t = 1$	o_{11}	o_{12}	=	a'_{11}	a'_{12}	a'_{13}	×	v_{11}	v_{12}	=	a'_{11} =0.6	×	v_{11}	v_{12}
										+				
										a'_{12} =0.3	×	v_{21}	v_{22}
										+				
										a'_{13} =0.1	×	v_{31}	v_{32}

第一個時間的輸出 = a'_{11} *第一個時間的value + a'_{12} *第2個時間的value + a'_{13} *第3個時間的value

當Attention Score為單位矩陣

$$\boldsymbol{O}_{T\times d}, \boldsymbol{Q}_{T\times d}, \boldsymbol{K}_{T\times d}, \boldsymbol{V}_{T\times d}$$

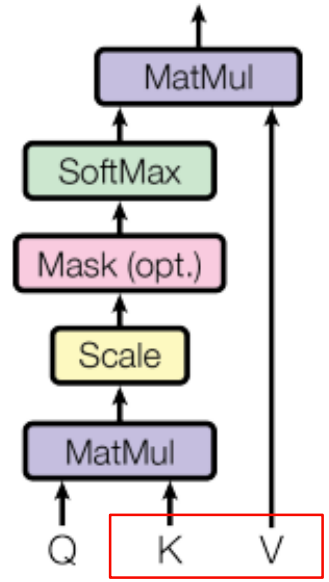
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

$$\boldsymbol{O} \approx \boldsymbol{Q}\boldsymbol{K}^{\boldsymbol{T}}\boldsymbol{V}$$

$$\boldsymbol{O}_{T_Q\times d} \quad \boldsymbol{A}_{T_Q\times T_K} \quad \boldsymbol{V}_{T\times d}$$

$$\begin{bmatrix} o_{11} & o_{12} & \cdots & o_{1d} \\ o_{21} & o_{22} & \cdots & o_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ o_{T1} & o_{d2} & \cdots & o_{Td} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1d} \\ v_{21} & v_{22} & \cdots & v_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ v_{T1} & v_{d2} & \cdots & v_{Td} \end{bmatrix}$$

$$O = V$$



第一個時間的輸出 = a'_{11} *第一個時間的value + a'_{12} *第2個時間的value + a'_{13} *第3個時間的value

= 1*第一個時間的value + 0*第2個時間的value + 0*第3個時間的value

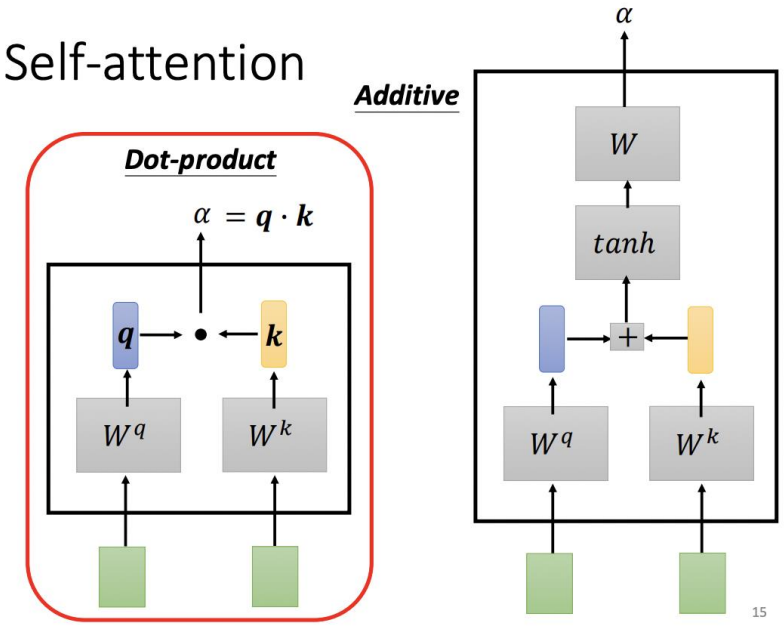
第一個時間的輸出 = 第一個時間的value

每個時間之間沒有關聯性= A單位矩陣

論文提到的Self-Attention

The two most commonly used attention functions are additive attention [2], and dot-product (multi-plicative) attention. Dot-product attention is identical to our algorithm, except for the scaling factor of $\frac{1}{\sqrt{d_k}}$. Additive attention computes the compatibility function using a feed-forward network with a single hidden layer. While the two are similar in theoretical complexity, dot-product attention is much faster and more space-efficient in practice, since it can be implemented using highly optimized matrix multiplication code.

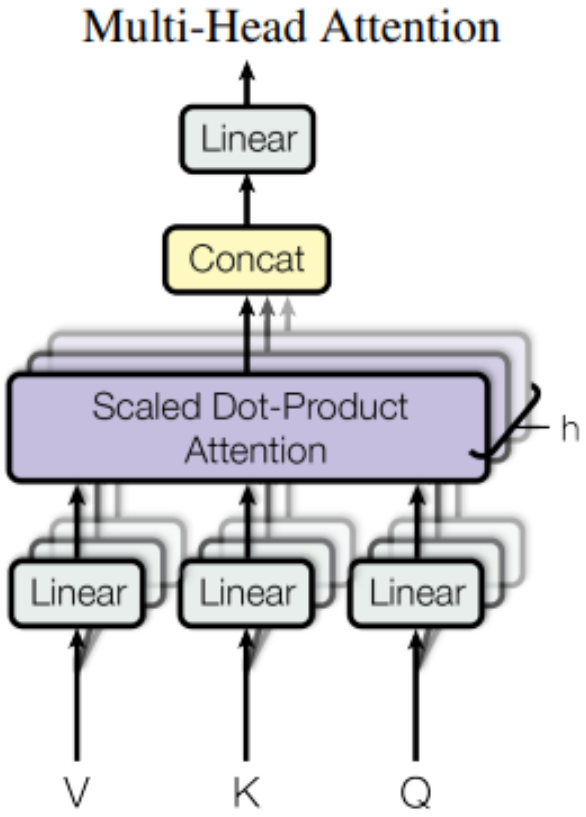
Self-attention

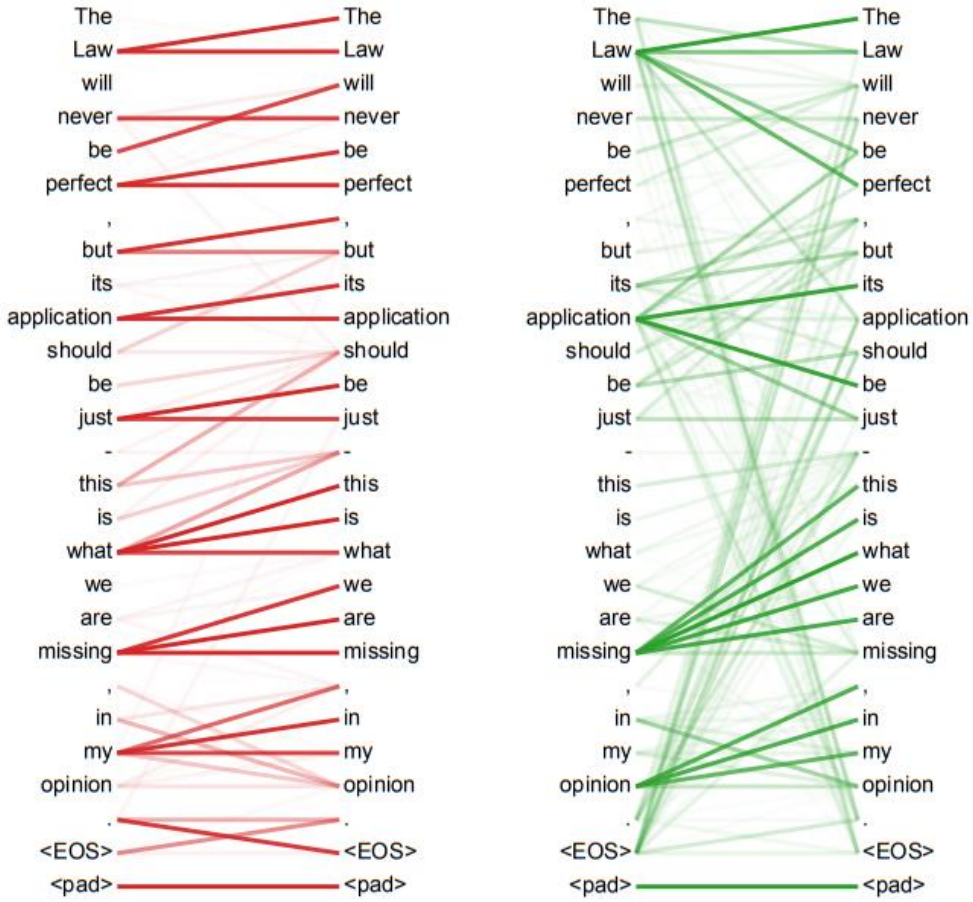


Reference: 李宏毅老師

Multi-head Self-attention

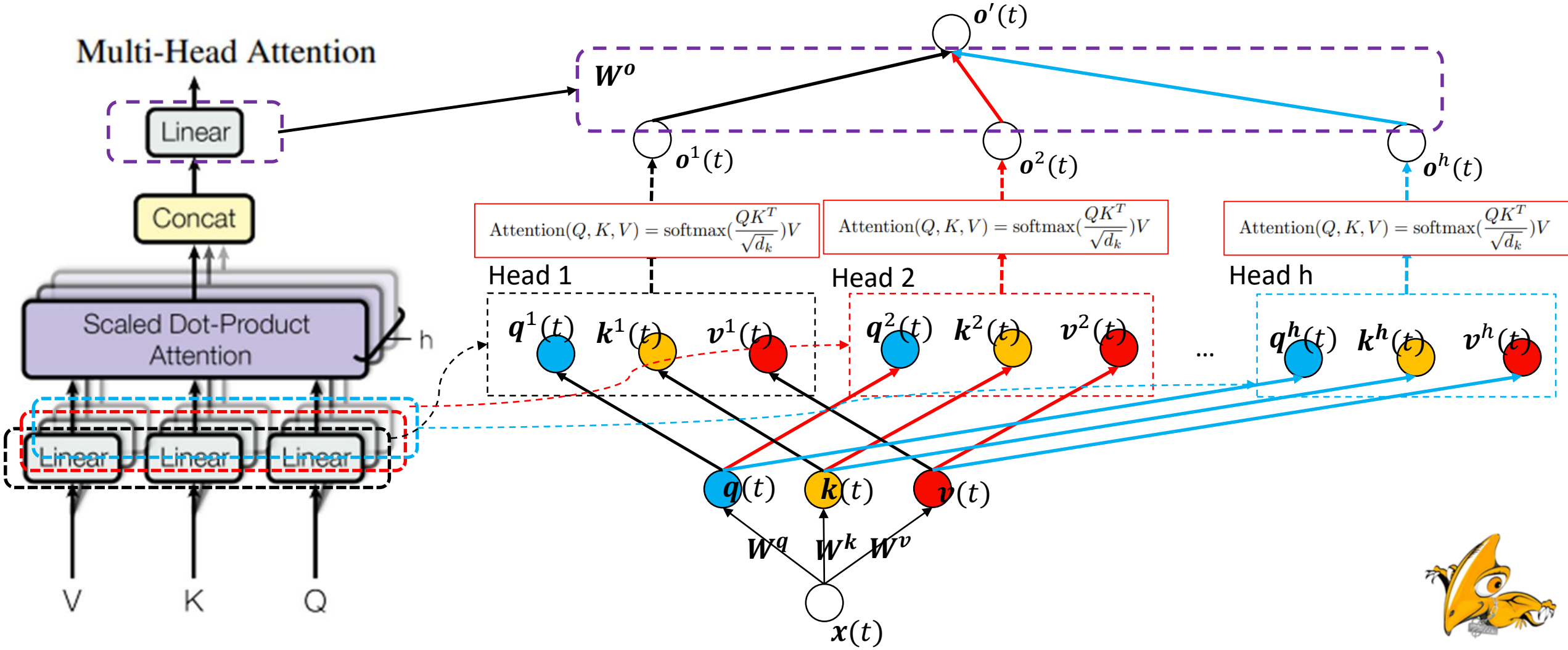
- 相關性在不同的位置可能有不同類別，所以需要多Multi-head來讓模型可以得到更多不同位置的表示。

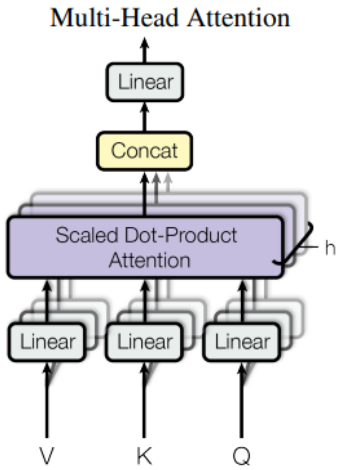




Many of the attention heads exhibit behaviour that seems related to the structure of the sentence.

Multi-head Self-attention





$$Q^{(h)}{}_{T\times d} = W^{(h)}{}_{q} Q_{T\times d}\,;\quad K^{(h)}{}_{T\times d} = W^{(h)}{}_{k} K_{T\times d}\,;\quad V^{(h)}{}_{T\times d} = W^{(h)}{}_{v} V_{T\times d}$$

範例有4個head→$h = 1, 2, 3, 4$

$X_{3\times 4}$

$W^{q}{}_{4\times 2}$ $W^{k}{}_{4\times 2}$ $W^{v}{}_{4\times 2}$

$Q_{3\times 2}$ $K_{3\times 2}$ $V_{3\times 2}$

$W^{(1)}{}_{q}$ $W^{(1)}{}_{k}$ $W^{(1)}{}_{v}$

$W^{(2)}{}_{q}$ $W^{(2)}{}_{k}$ $W^{(2)}{}_{v}$

$W^{(3)}{}_{q}$ $W^{(3)}{}_{k}$ $W^{(3)}{}_{v}$

$W^{(4)}{}_{q}$ $W^{(4)}{}_{k}$ $W^{(4)}{}_{v}$

$Q^{(1)}$ $K^{(1)}$ $V^{(1)}$ → ***self attention*** (1) → $O^{(1)}{}_{3\times 2}$

$Q^{(2)}$ $K^{(2)}$ $V^{(2)}$ → ***self attention*** (2) → $O^{(2)}{}_{3\times 2}$

$Q^{(3)}$ $K^{(3)}$ $V^{(3)}$ → ***self attention*** (3) → $O^{(3)}{}_{3\times 2}$

$Q^{(4)}$ $K^{(4)}$ $V^{(4)}$ → ***self attention*** (4) → $O^{(4)}{}_{3\times 2}$

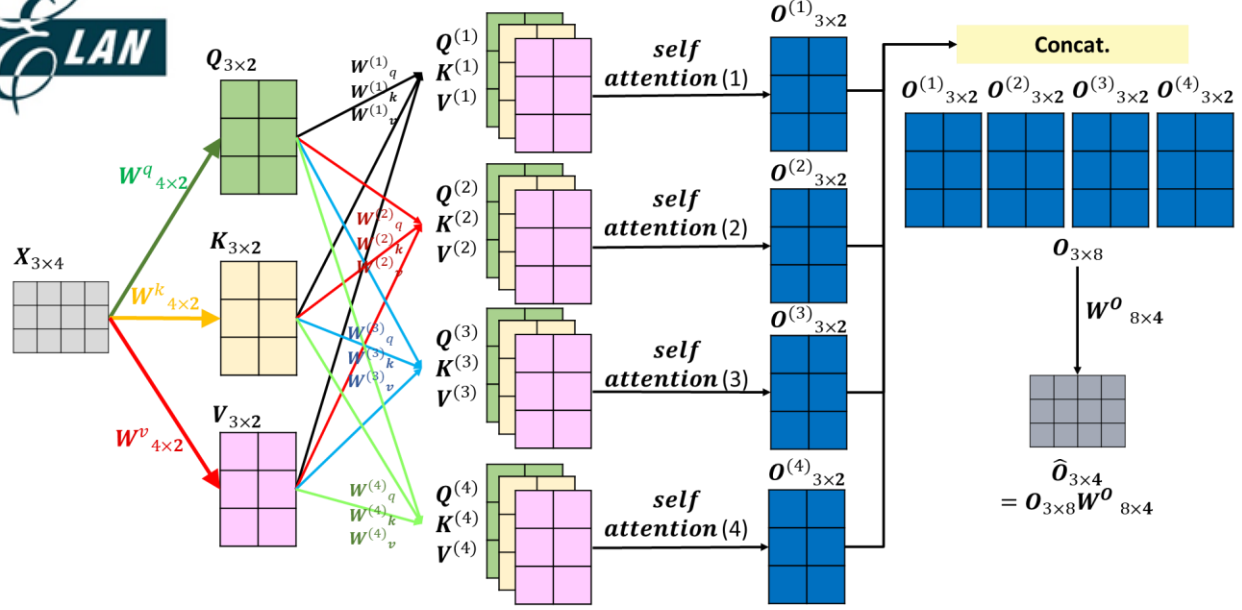
Concat.

$O^{(1)}{}_{3\times 2}$ $O^{(2)}{}_{3\times 2}$ $O^{(3)}{}_{3\times 2}$ $O^{(4)}{}_{3\times 2}$

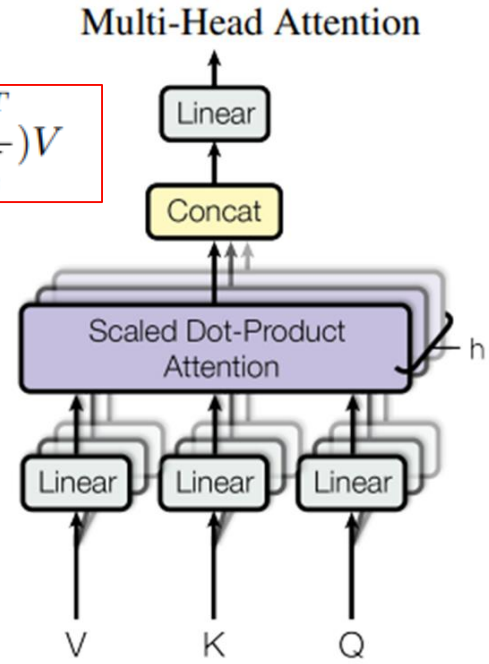
$O_{3\times 8}$

$W^{O}{}_{8\times 4}$

$\hat{O}_{3\times 4} = O_{3\times 8} W^{O}{}_{8\times 4}$



$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



參數初估

Multi-Head Cross Attention:

$$\boldsymbol{X}_{T\times dx}, \boldsymbol{Q}_{T\times d}, \boldsymbol{K}_{T\times d}, \boldsymbol{V}_{T\times d}, \boldsymbol{O}_{T\times dx}$$

$$\boldsymbol{W^q}, \boldsymbol{W^k}, \boldsymbol{W^v} \in \mathbb{R}^{dx\times d}$$

$$\boldsymbol{W}^{(h)}{}_{\boldsymbol{q}}, \boldsymbol{W}^{(h)}{}_{\boldsymbol{k}}, \boldsymbol{W}^{(h)}{}_{\boldsymbol{v}} \in \mathbb{R}^{T\times d}, \qquad \boldsymbol{W^O} \in \mathbb{R}^{(n_h\times d)\times dx}$$

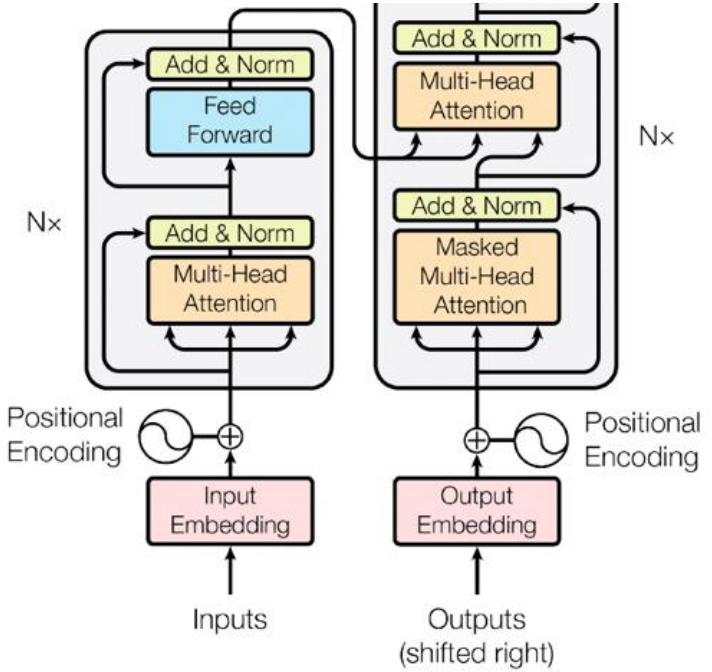
$$\boldsymbol{W^q} + \boldsymbol{W^k} + \boldsymbol{W^v} + \boldsymbol{W^O} + \sum_{\boldsymbol{h}=1}^{n_h} \boldsymbol{W}^{(h)}{}_{\boldsymbol{q}} + \boldsymbol{W}^{(h)}{}_{\boldsymbol{k}} + \boldsymbol{W}^{(h)}{}_{\boldsymbol{v}}$$

$$T = 10, dx = 300, d = 300, n_h = 8$$

$$dx \times d \times 3 + n_h \times (T \times d \times 3) + (n_h + d) \times dx$$

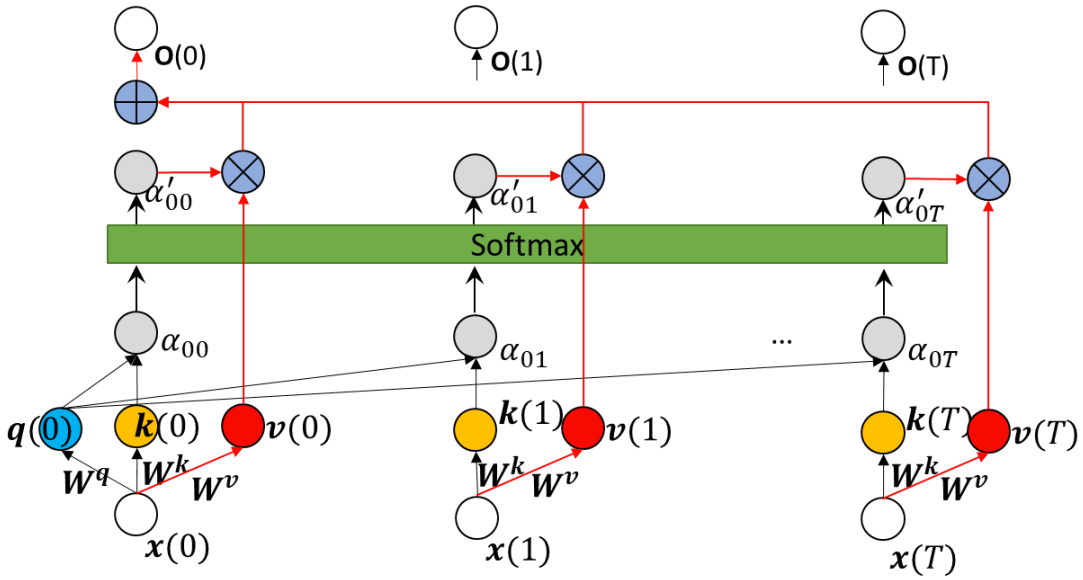
$$= 300 \times 300 \times 3 + 8 \times (10 \times 300 \times 3) + (8 + 300) \times 300$$

$$= 434{,}400$$



Positional Encoding

輸入是有訊續的$\boldsymbol{x}(0), \boldsymbol{x}(1), \ldots, \boldsymbol{x}(T)$，但在前面介紹的Attention是沒有考慮順序性的。



因為架構沒有順序(位置)資料，所以在Transformer中每一個位置給予一個positional vector($\boldsymbol{e}_i, \forall i = 0,1, \ldots, T$)，整個過程稱為position encoding。

Positional Encoding

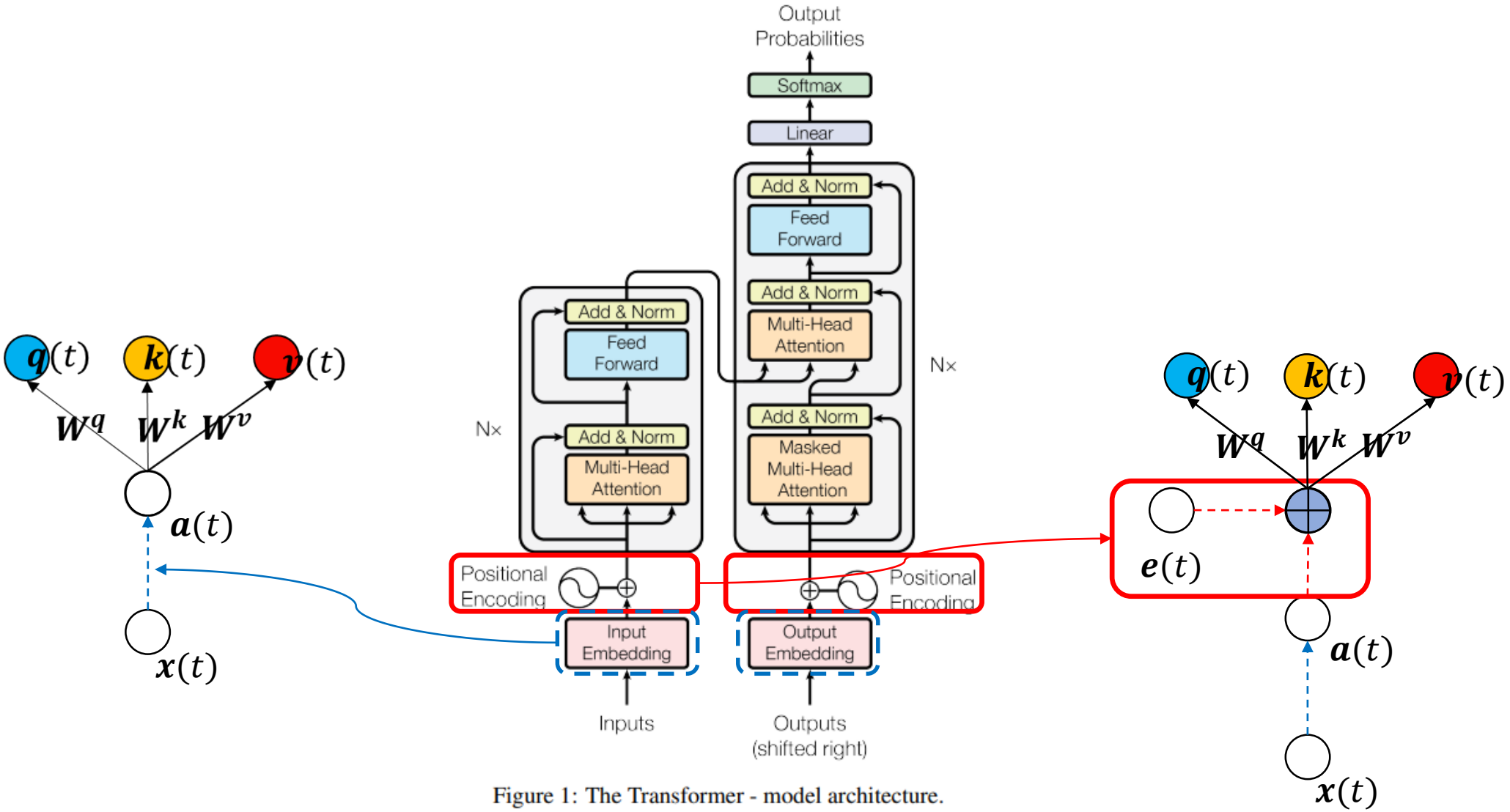
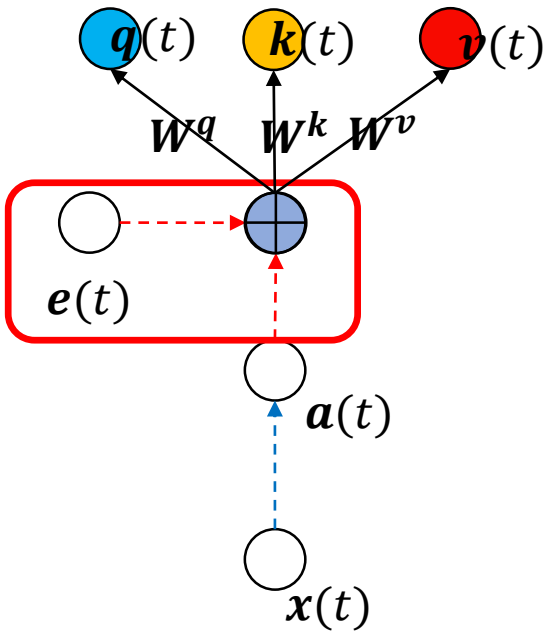


Figure 1: The Transformer - model architecture.

Positional Encoding



$$a(1) + e(1) = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}_{t=1} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}_{t=1}$$

$$e(1) = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}_{t=1} \quad a(1) = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}_{t=1}$$

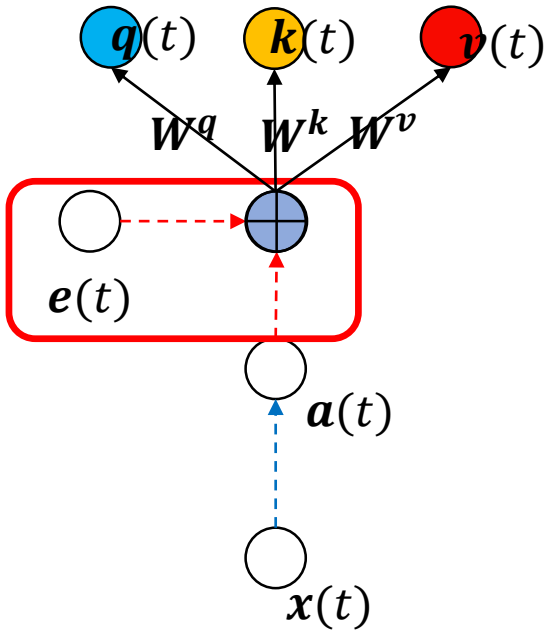
$$a(2) + e(2) = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}_{t=2} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}_{t=2}$$

$$e(2) = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}_{t=1} \quad a(2) = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}_{t=2}$$

$$a(3) + e(3) = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}_{t=3} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}_{t=3}$$

$$e(3) = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}_{t=1} \quad a(T=3) = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}_{t=3}$$

Positional Encoding



$$a(1) + e(1) = \begin{bmatrix} 1.1 \\ 0.2 \\ 0.3 \\ 0.4 \end{bmatrix}$$

$$e(1) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad a(1) = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.3 \\ 0.4 \end{bmatrix}$$

$$a(2) + e(2) = \begin{bmatrix} 0.2 \\ 1.3 \\ 0.2 \\ 0.3 \end{bmatrix}$$

$$e(2) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad a(2) = \begin{bmatrix} 0.2 \\ 0.3 \\ 0.2 \\ 0.3 \end{bmatrix}$$

$$a(3) + e(3) = \begin{bmatrix} 0.1 \\ 0.1 \\ 1.1 \\ 0.1 \end{bmatrix}$$

$$e(3) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad a(3) = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \\ 0.1 \end{bmatrix}$$

Add & Norm

Multi-Head Attention

$$\boldsymbol{O} = \text{MultiHeadAttent}(\boldsymbol{X})$$

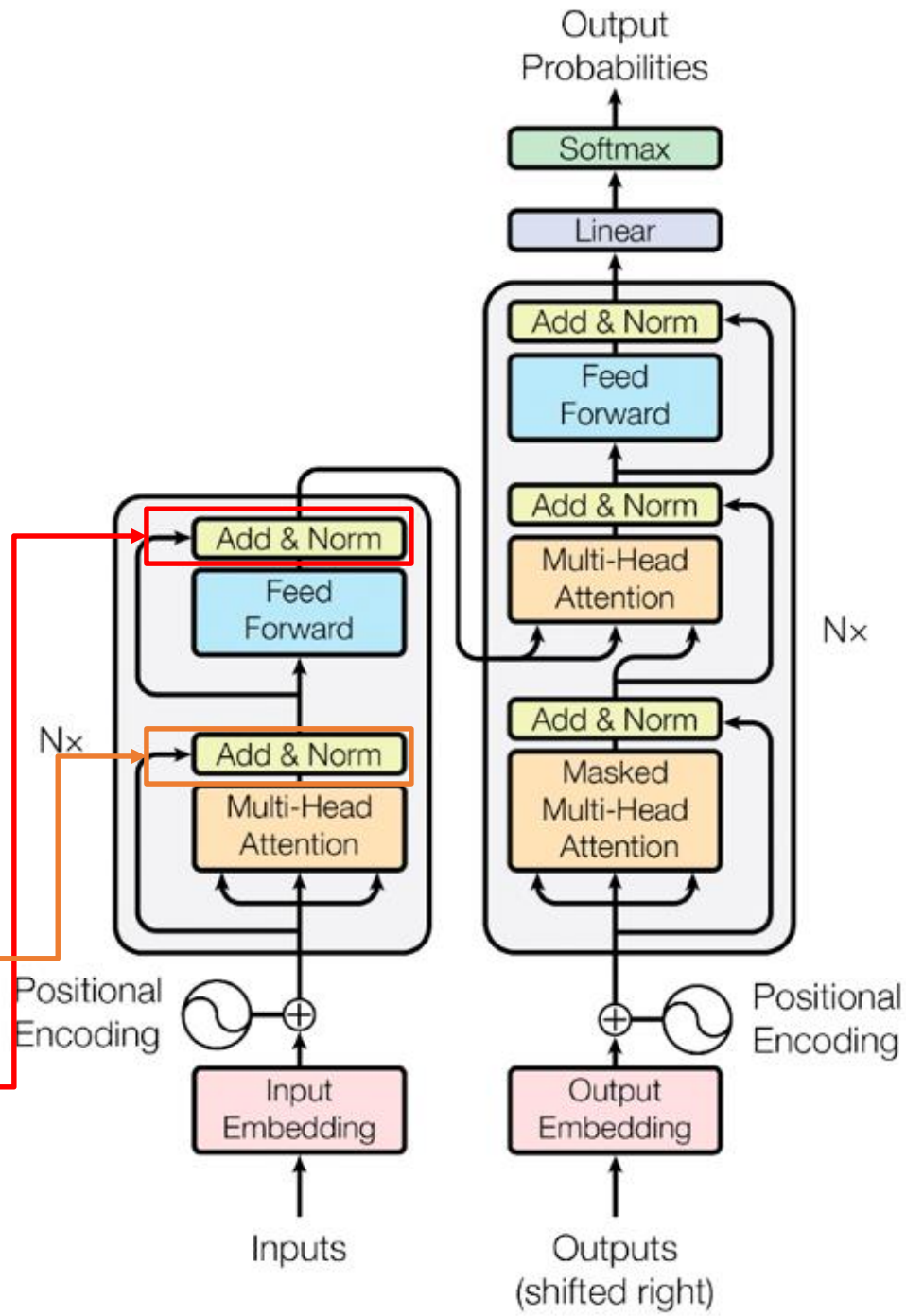
Add

$$\boldsymbol{X} + \text{MultiHeadAttent}(\boldsymbol{X})$$

Add & Norm

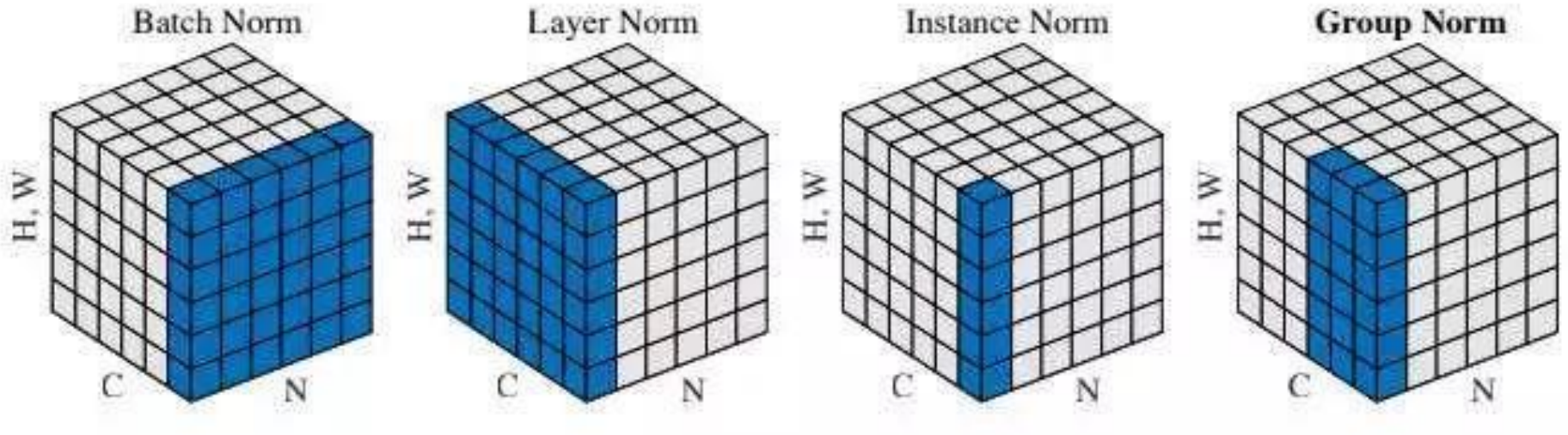
$$\text{LayerNorm}(\boldsymbol{X} + \text{MultiHeadAttent}(\boldsymbol{X}))$$

$$\text{LayerNorm}(\boldsymbol{X} + \mathit{FeedForward}(\boldsymbol{X}))$$

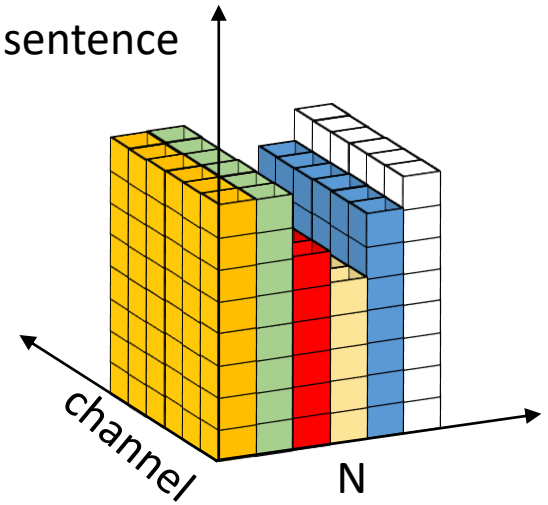


Normalization

$$y = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta$$



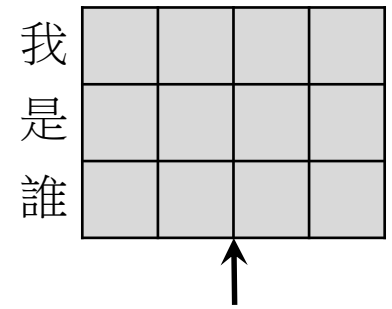
BN: 針對每個channel的batch data所有分布來計算



語言模型的句子長度不固定
(每筆資料的Z軸實際大小不一)
用Layer Norm會比較合適。

常用於RNN結構內

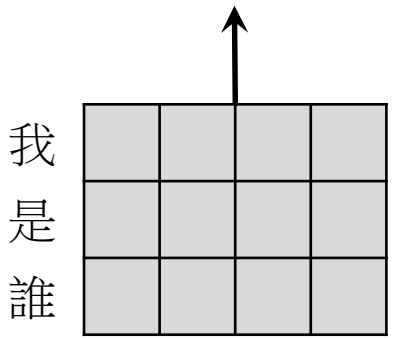
Encoder Block



輸出的encode matrix=Encode features

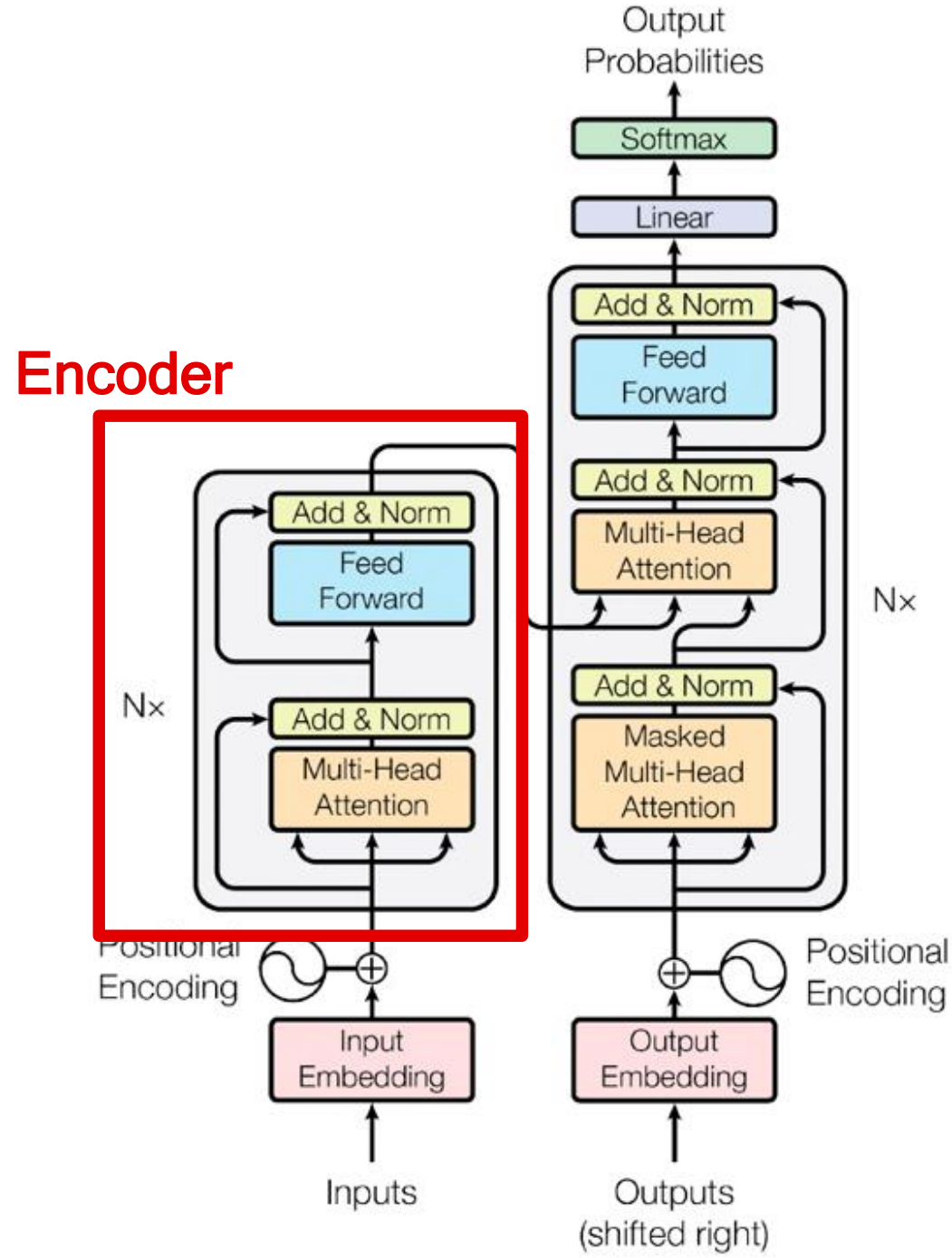
$\hat{O}_{3\times4}$

N×Encoder



輸入Data Matrix

$X_{3\times4}$



Decoder Block

兩個Multi-Head Attention

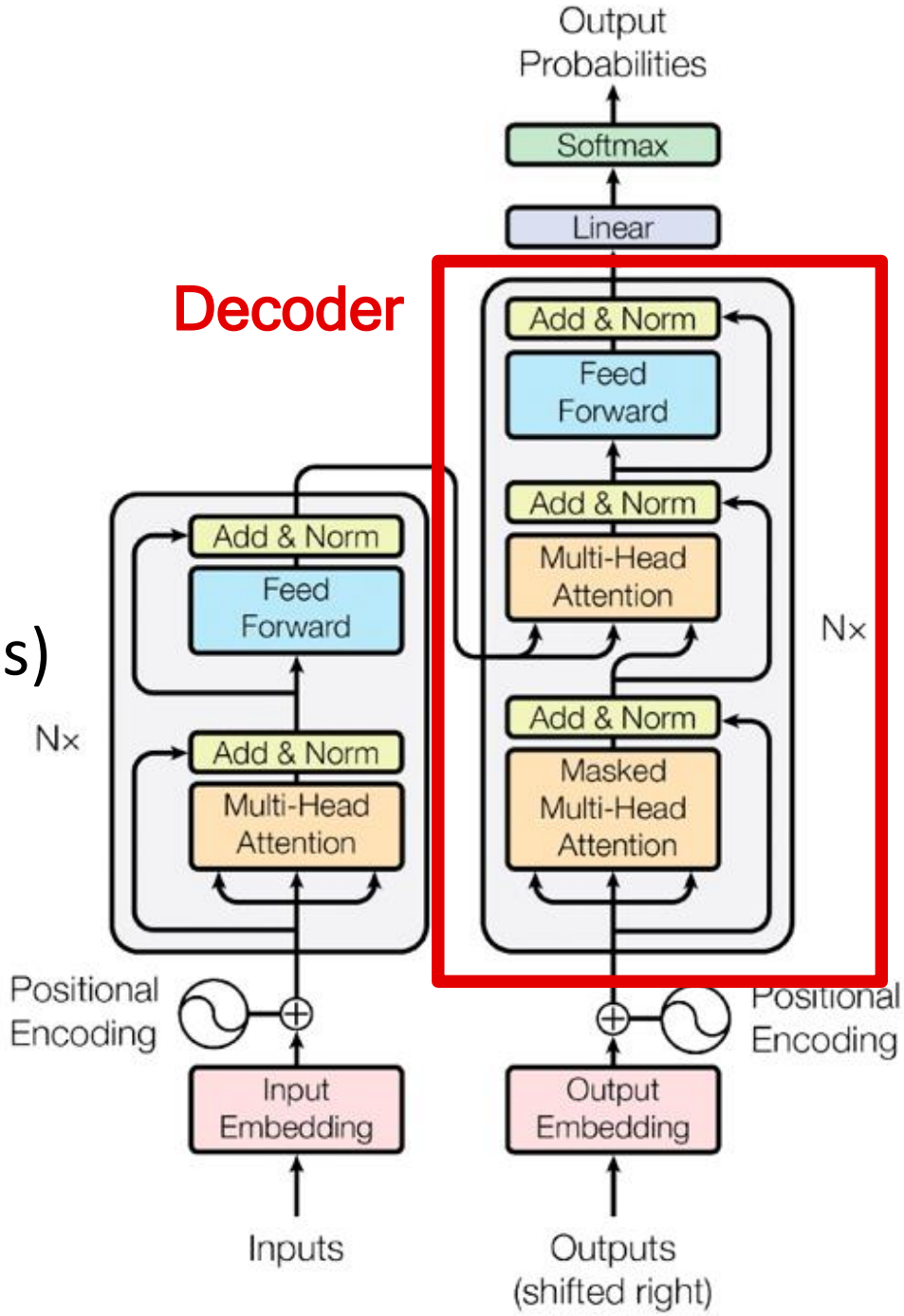
1. Masked Multi-Head Attention

2. Multi-Head Attention (包含Encode的features)

Q: Masked Multi-Head Attention 的輸出

K: Encode的features

V: Encode的features

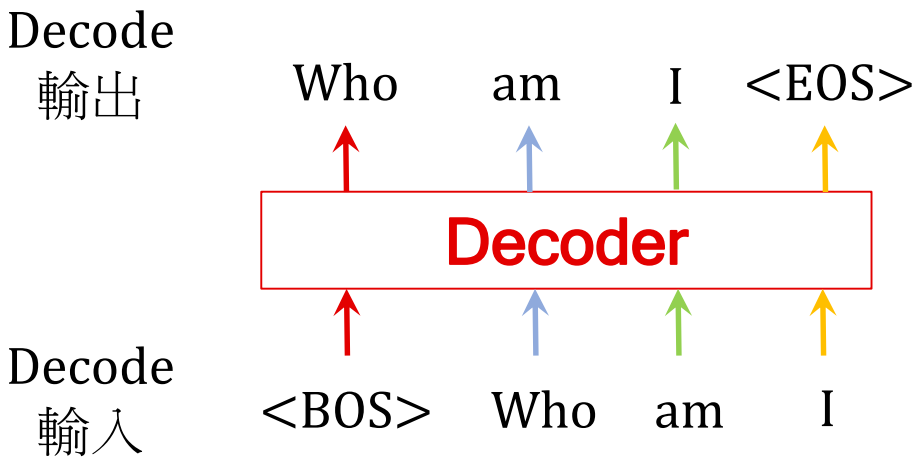


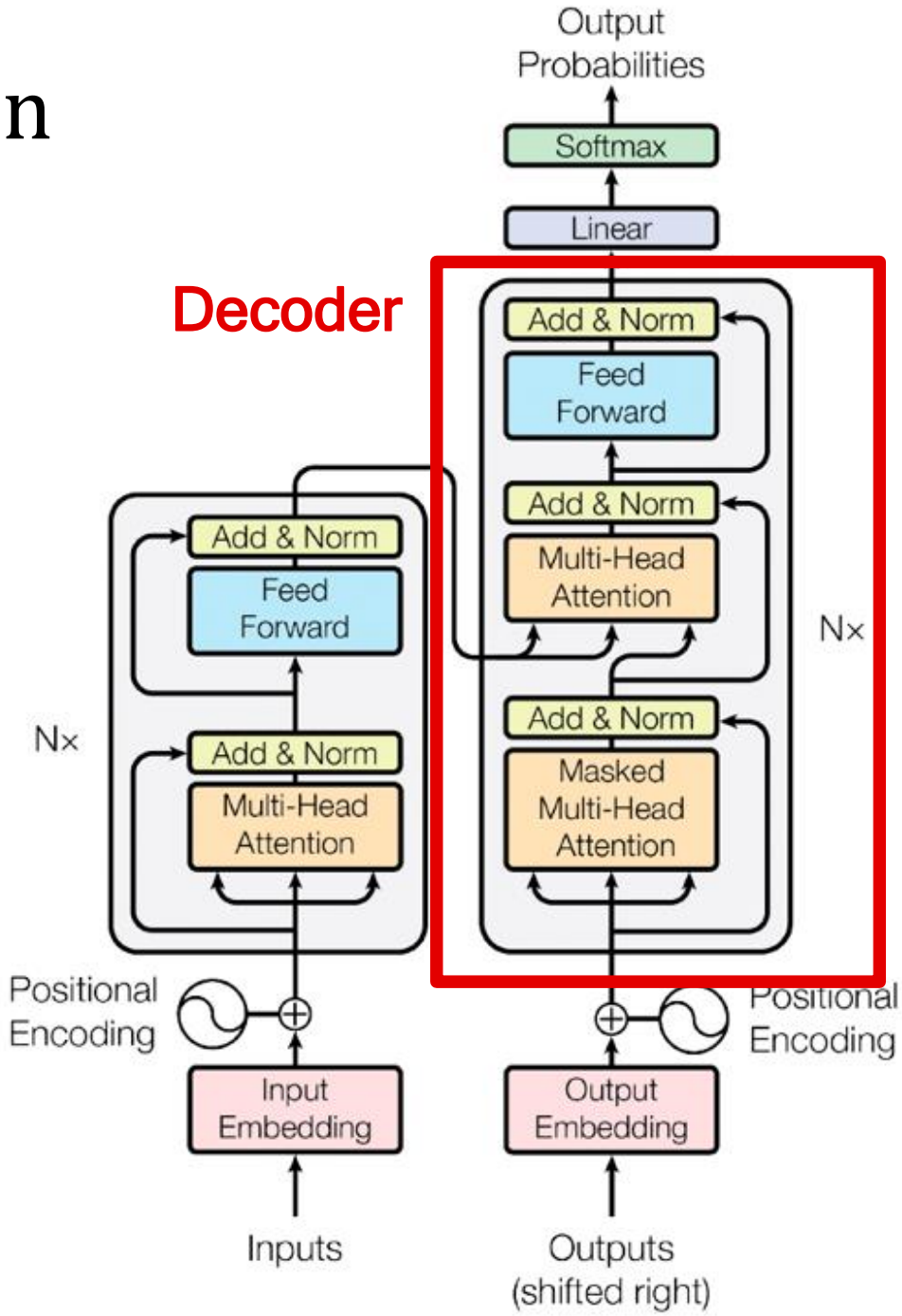
1. Masked Multi-Head Attention

Hugging Face is a startup based in New York City and Paris

p(word)

https://huggingface.co/docs/transformers/perplexity





Context1:

<BOS> I am Tommy <EOS>

1. <BOS> → I
2. <BOS> I → am
3. <BOS> I am → Tommy
4. <BOS> I am Tommy →<EOS>

還有很多組合

Context2:

<BOS> You are fine <EOS>

1. <BOS> → You
2. <BOS> You → are
3. <BOS> You are → fine
4. <BOS> You are fine →<EOS>

N-gram $p(x_1, x_2, \dots, x_L) = \prod_{i=1}^{L} p(x_i|x_1, \dots, x_{i-1}) \approx p(x_i|x_1, \dots, x_{i-1})$

N=1 $p(x_1, x_2, \dots, x_L) = \prod_{i=1}^{L} p(x_i|x_1, \dots, x_{i-1}) \approx p(x_i)$

N=2 $p(x_1, x_2, \dots, x_L) = \prod_{i=1}^{L} p(x_i|x_1, \dots, x_{i-1}) \approx p(x_i|x_{i-1})$

N=3 $p(x_1, x_2, \dots, x_L) = \prod_{i=1}^{L} p(x_i|x_1, \dots, x_{i-1}) \approx p(x_i|x_{i-1}, x_{i-2})$

以2-gram為例

Context1:

<BOS> I am Tommy <EOS>

↓

1. <BOS> → I
2. I → am
3. am → Tommy
4. Tommy →<EOS>

Context2:

<BOS> You are fine <EOS>

↓

1. <BOS> → You
2. You → are
3. are → fine
4. fine →<EOS>

以3-gram為例

Context1:

<BOS> I am Tommy <EOS>

↓

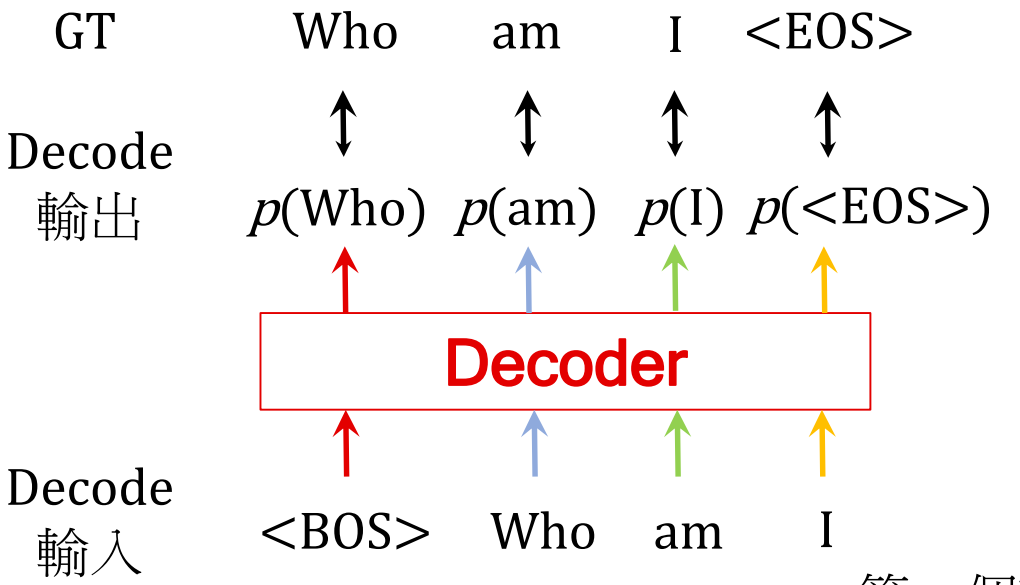
1. <BOS> I → am
2. I am→ Tommy
3. am Tommy → <EOS>

Context2:

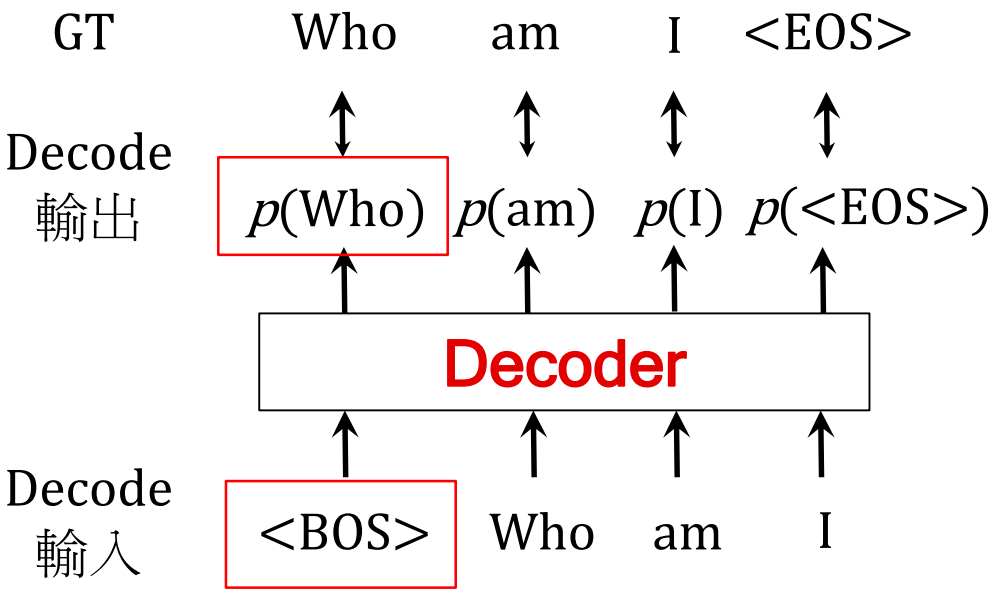
<BOS> You are fine <EOS>

↓

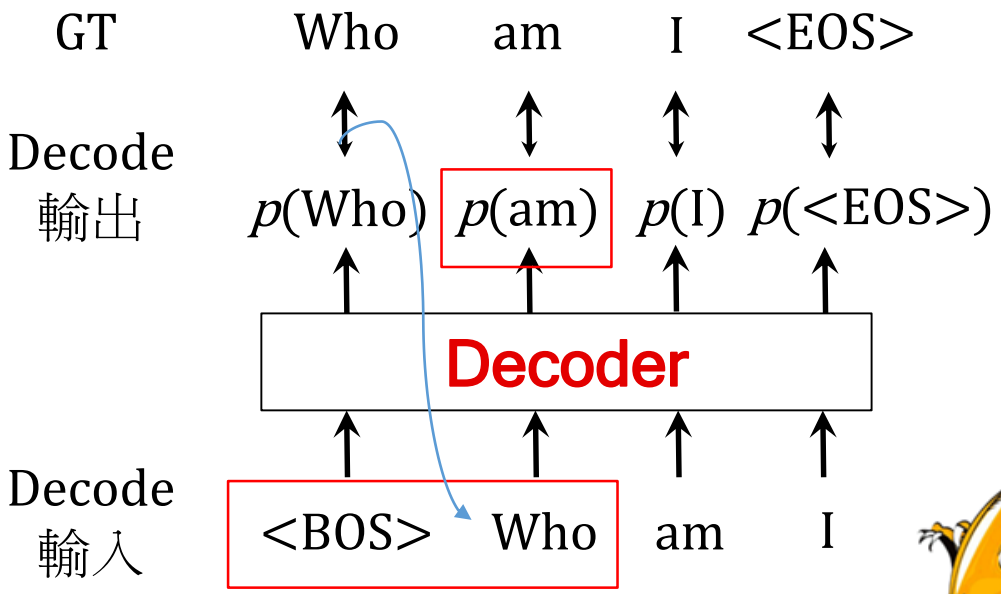
1. <BOS> You → are
2. You are → fine
3. are fine → <EOS>



第一個輸入
→第一個輸出

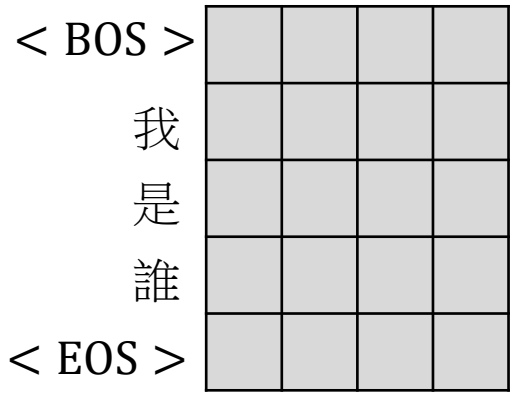


第一個輸入+第一個輸出
→第二個輸出

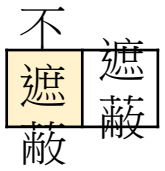


1. Masked Multi-Head Attention

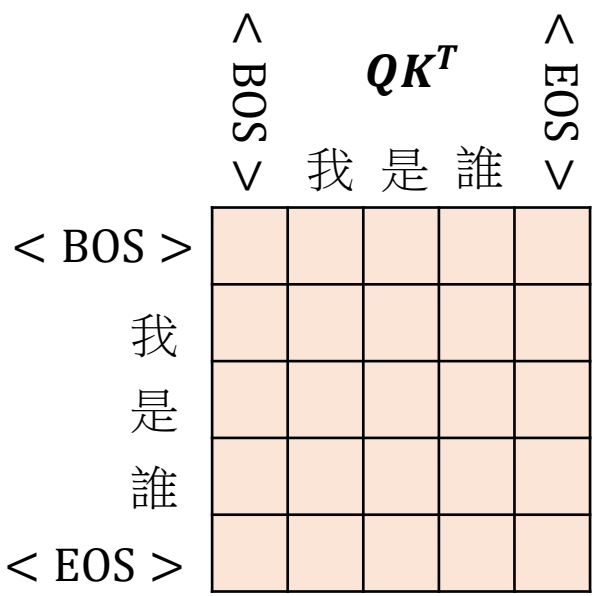
我是誰應該要寫Who am I
但不好呈現，所以我用中文取代



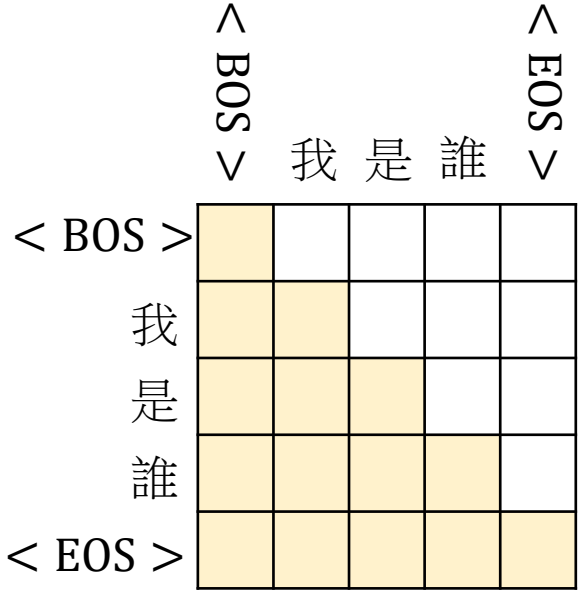
輸入矩陣X



得到Mask $\boldsymbol{QK^T}$後和$\boldsymbol{V}$內積後
就可以繼續算輸出的$\boldsymbol{O}$

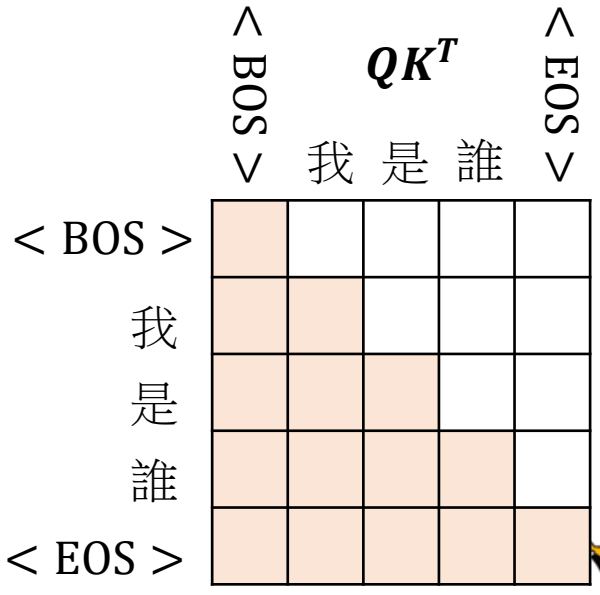


$\odot$



Mask矩陣

=

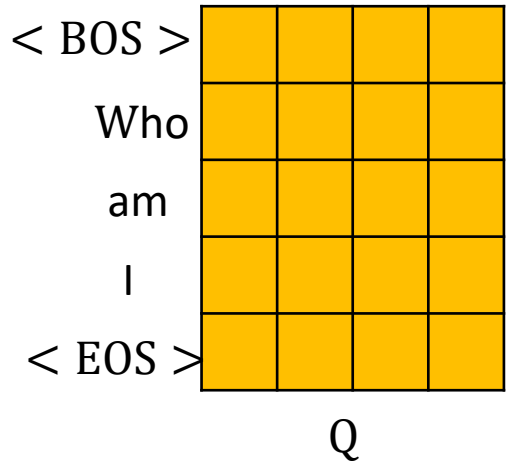


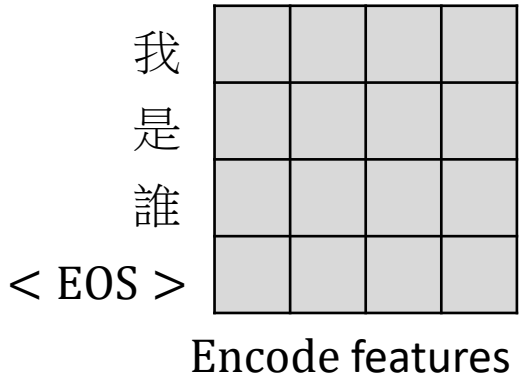
Mask $\boldsymbol{QK^T}$

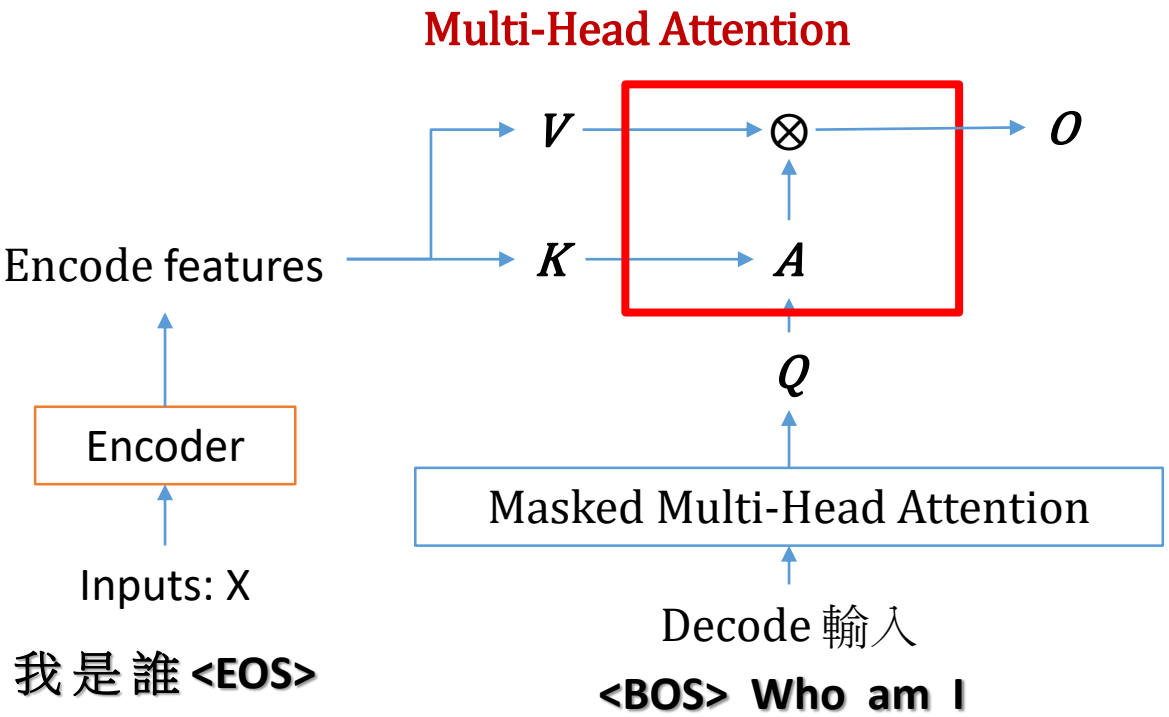
⊙ Hadamard product

2. Multi-Head Attention (包含Encode的features)

- Q: Masked Multi-Head Attention 的輸出
- K: Encode的features
- V: Encode的features







$O_{T\times d}$: 將V從K空間投影到Q空間

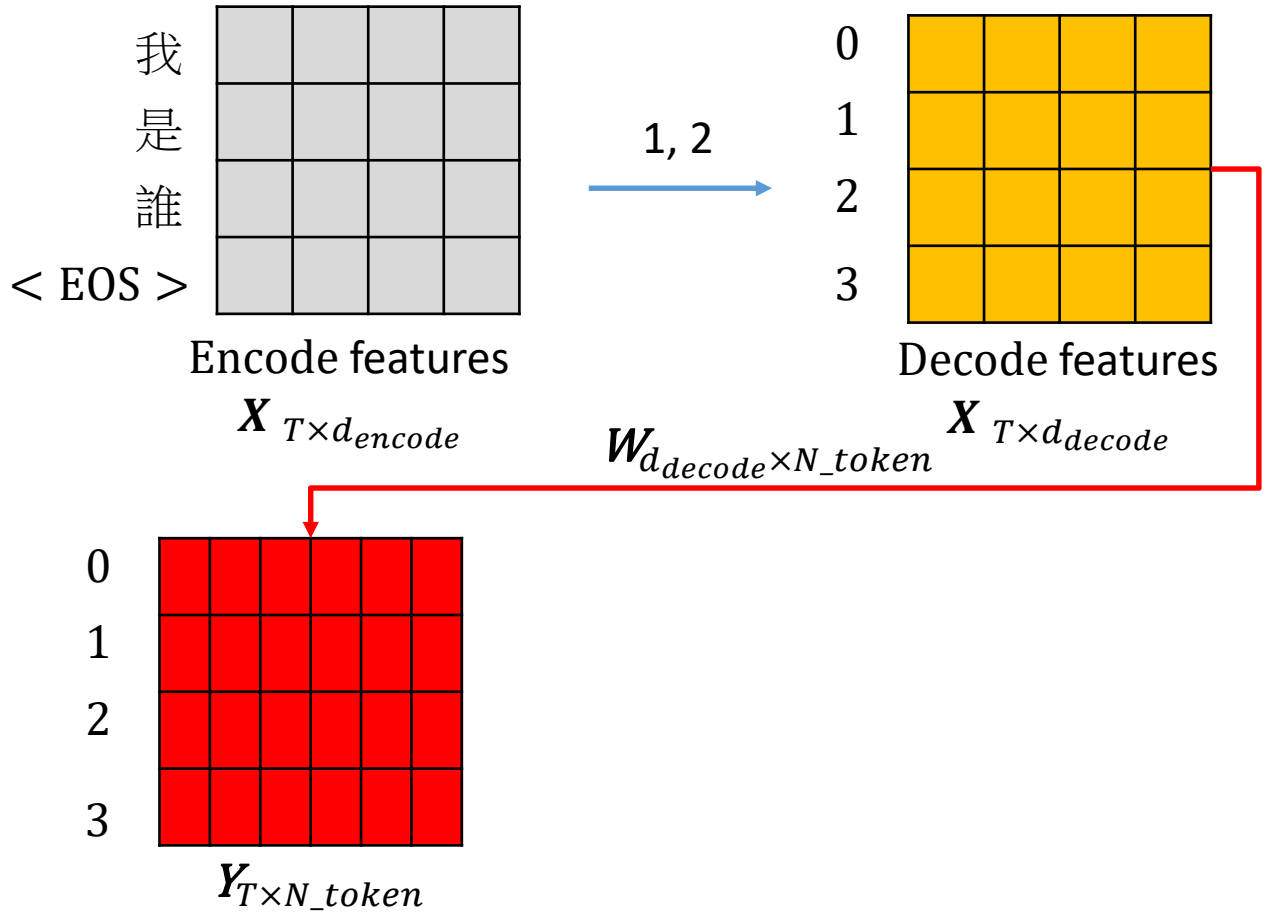
所以將Encode feature投影到Q空間

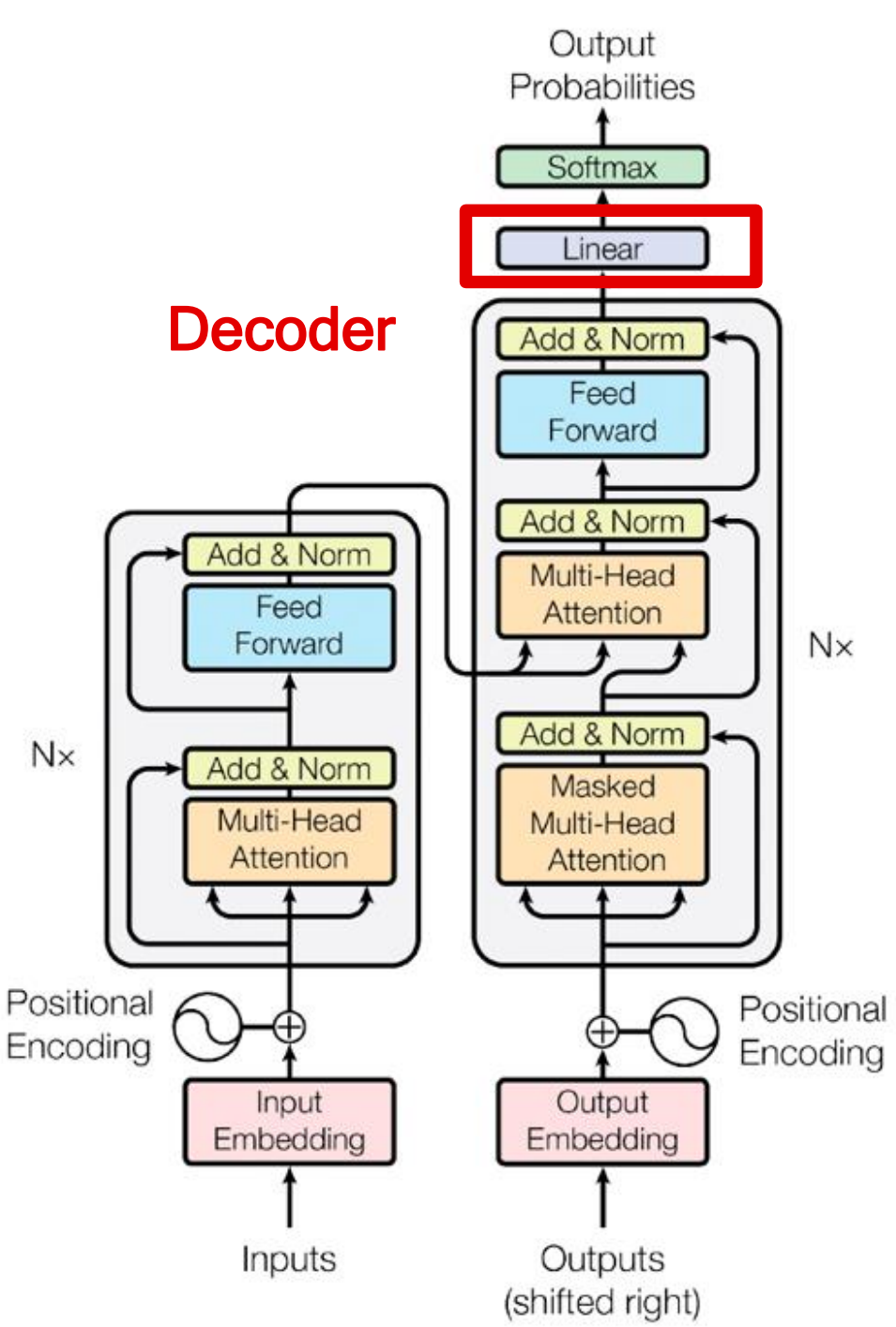
將中文的特徵投影到英文的特徵空間

Decoder Block

兩個Multi-Head Attention

1. Masked Multi-Head Attention
2. Multi-Head Attention (包含Encode的features)

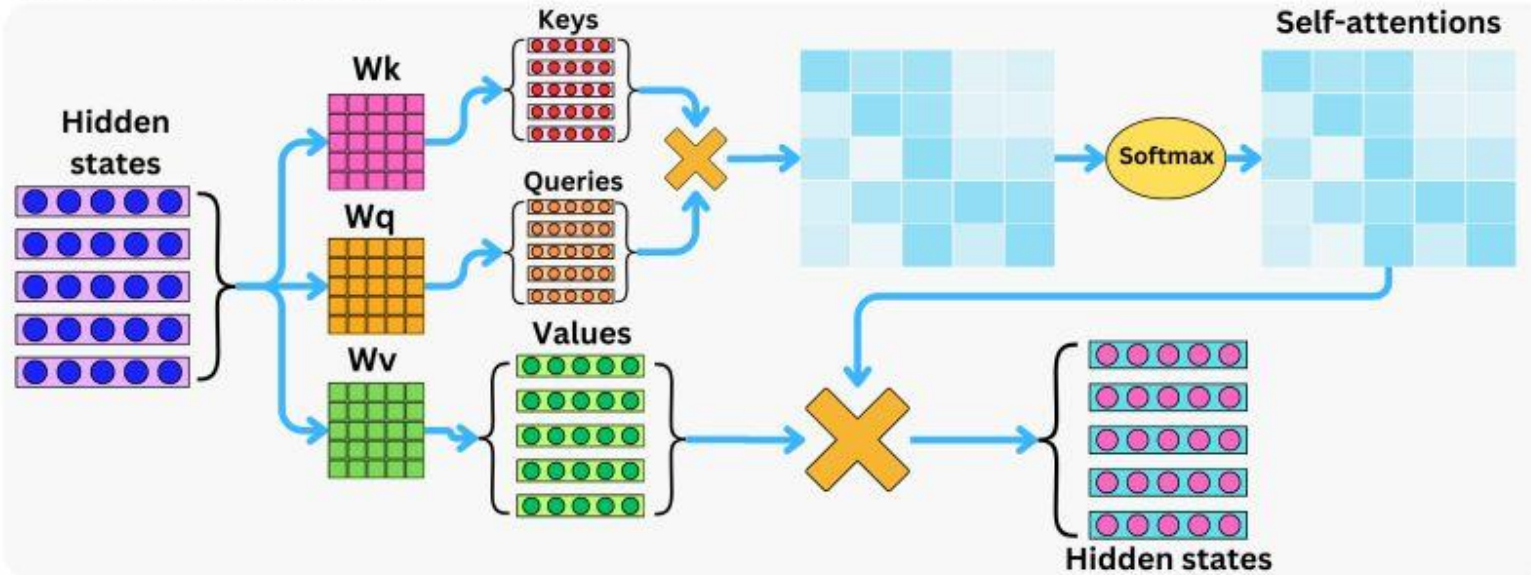




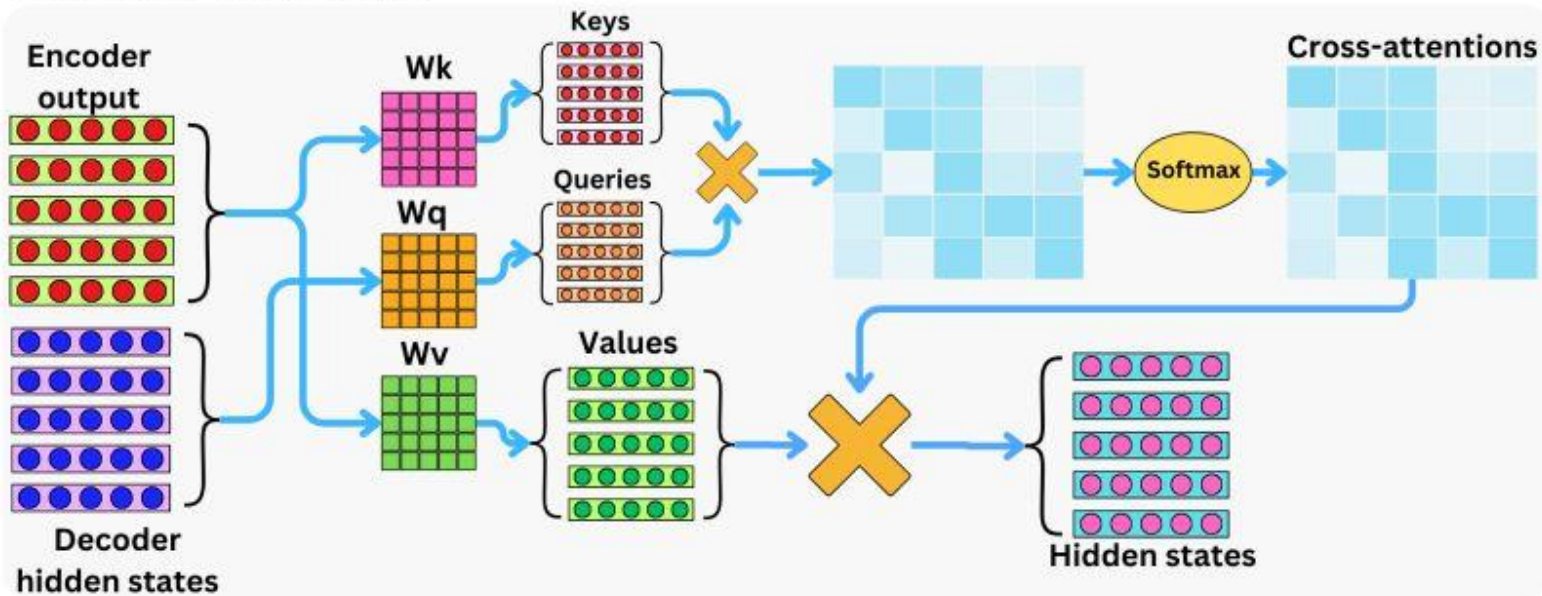
Self-Attention VS Cross-Attention

TheAiEdge.io

The Self-Attentions



The Cross-Attentions



Cross-attention

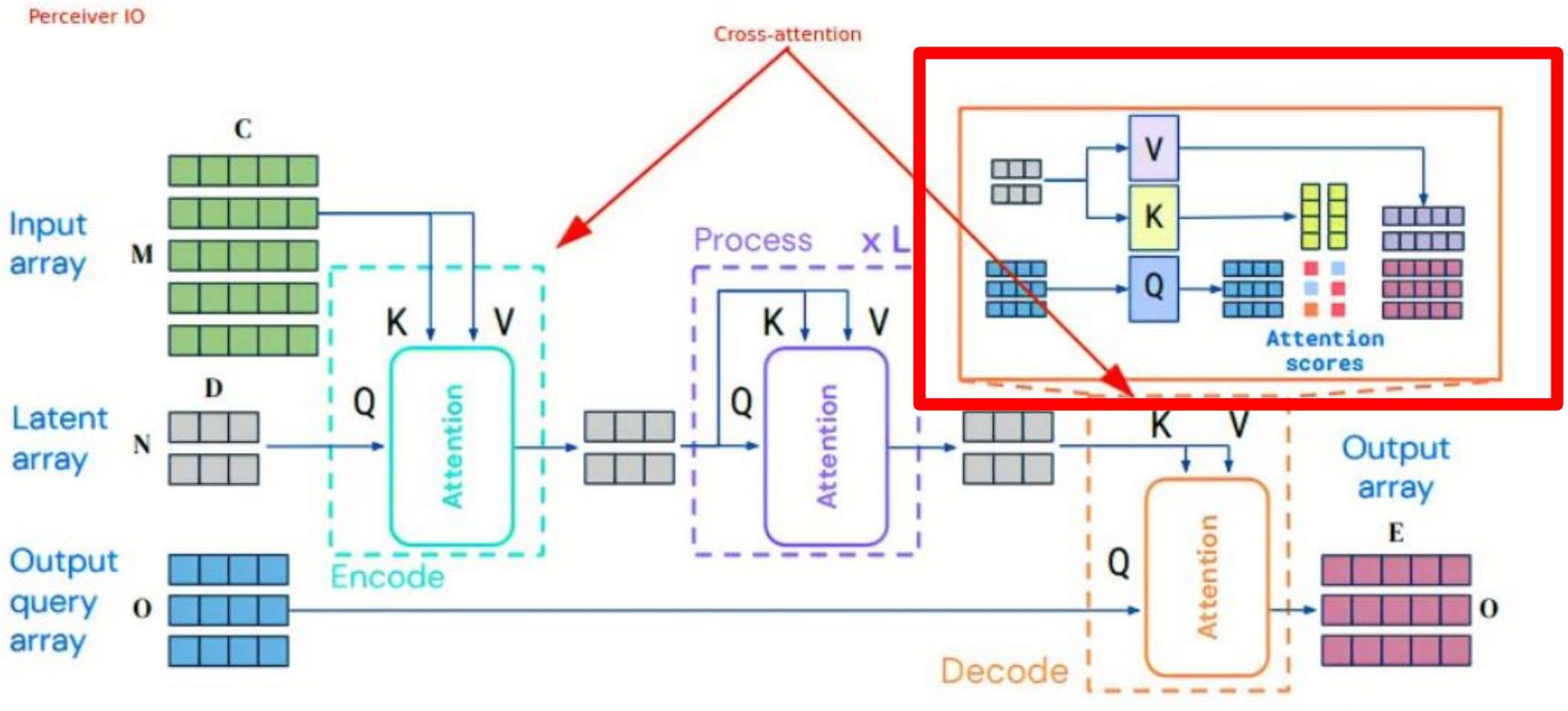
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

將V從K空間投影到Q空間

K空間: 文字　Q空間: 影像

K空間: Sensor 1　Q空間: Sensor 2

Self-attention: 輸入只有同一個序列資料
Cross-attention: 輸入來至不同的序列資料



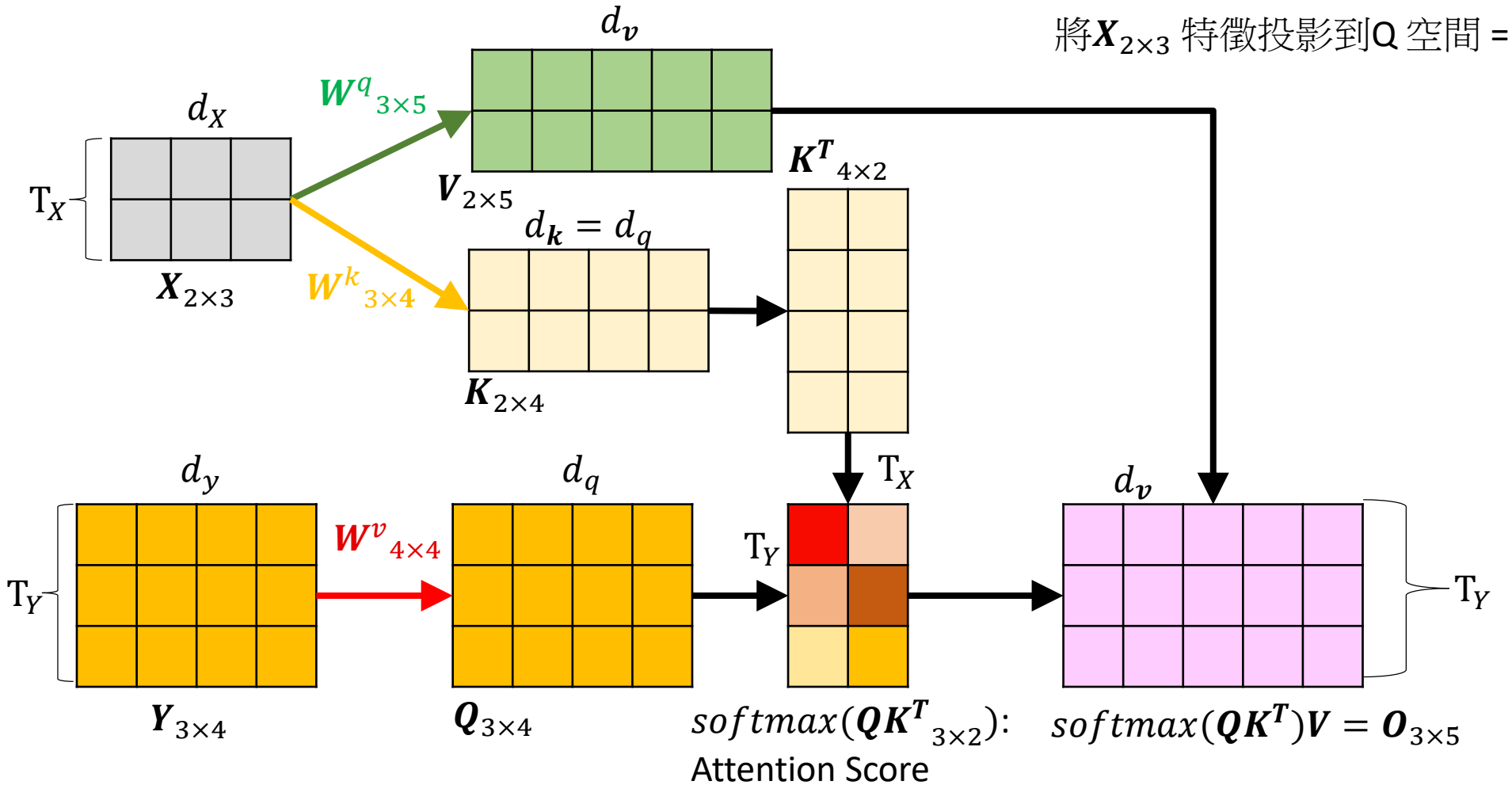
Cross-attention

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

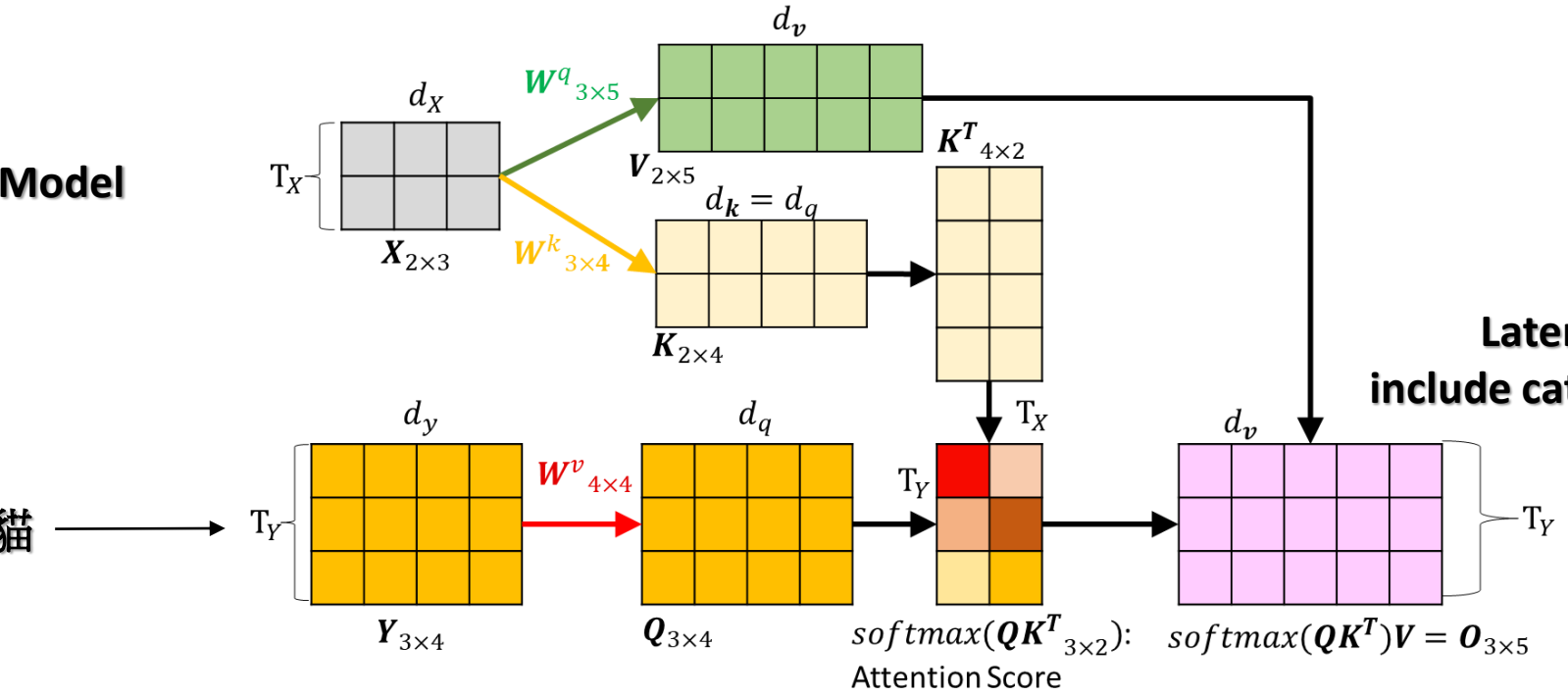
兩個不同的輸入源: $\boldsymbol{X}_{2\times3}$ $\boldsymbol{Y}_{3\times4}$
輸出的序列
長度會跟$\boldsymbol{Y}$ 一致
維度會跟V一致(可在投影時控制)

將V從K空間投影到Q空間

將$\boldsymbol{X}_{2\times3}$ 特徵投影到Q 空間 = $\boldsymbol{O}_{3\times5}$

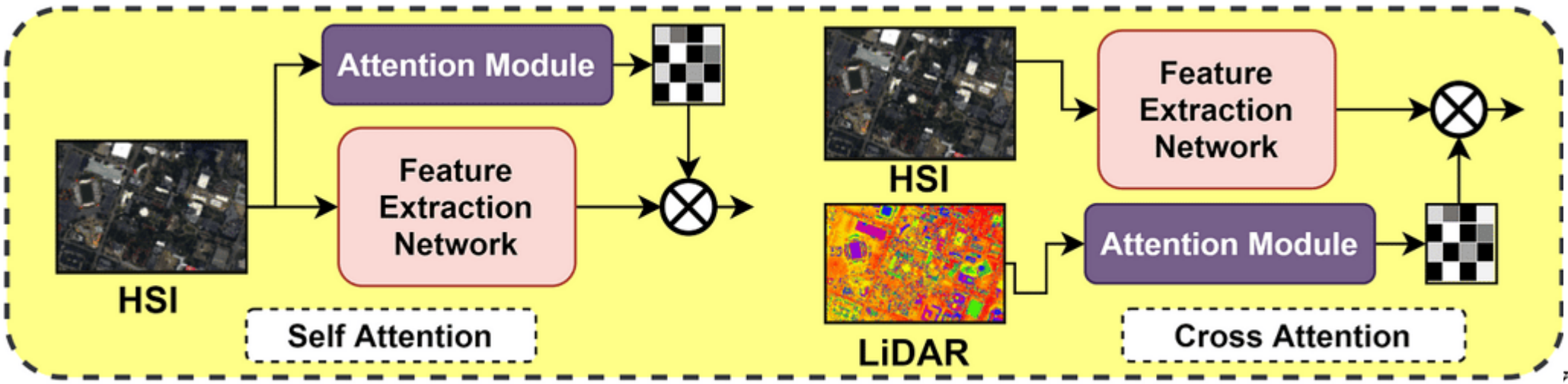


Diffusion Model

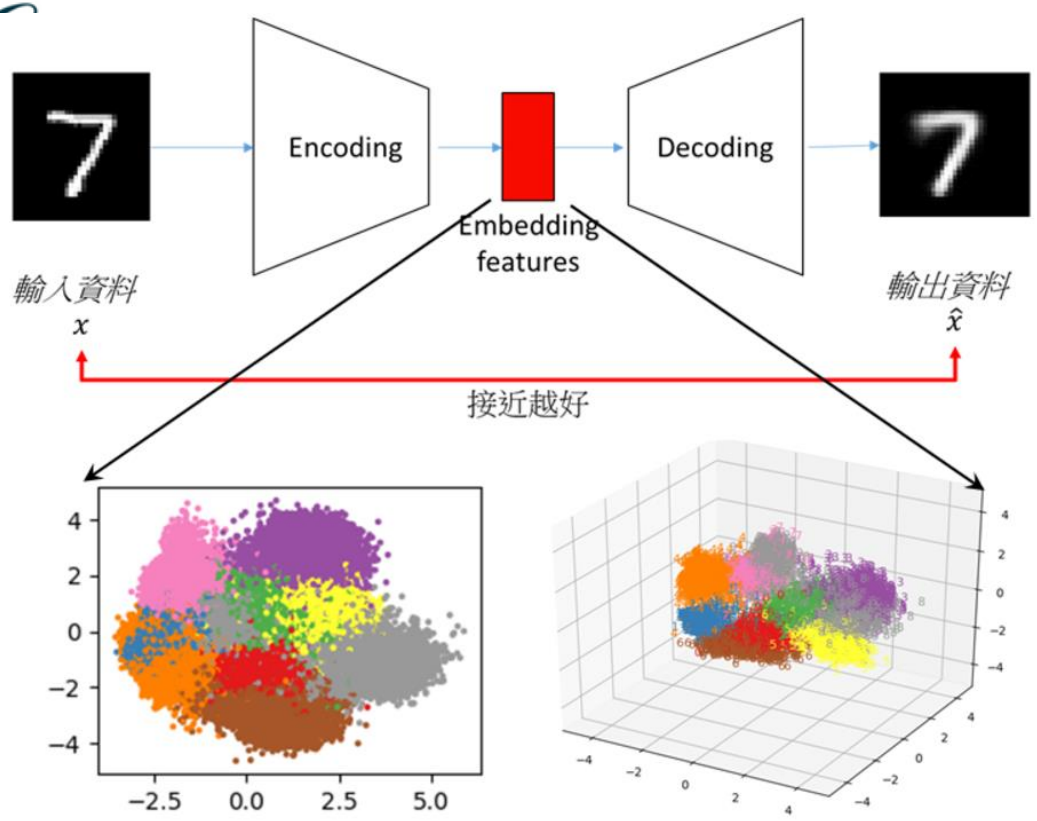


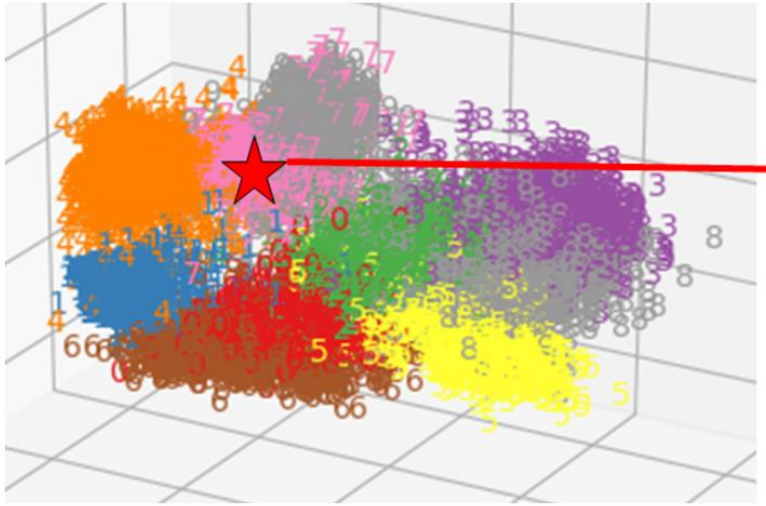
Latent array include cat information.

幫我生成一隻貓 →

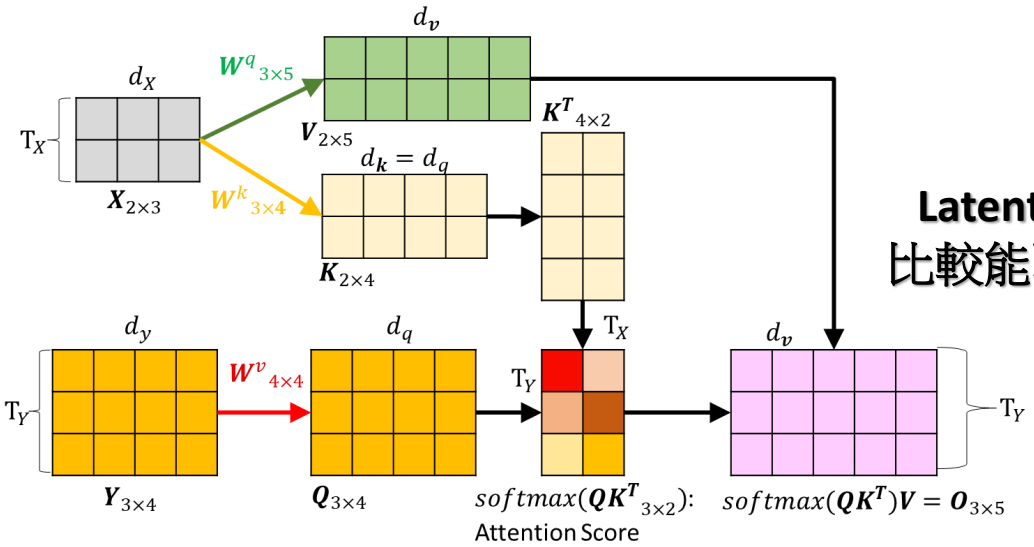


https://www.genspark.ai/spark/cross-attention-in-transformer-architecture/58187f91-5956-3099-a0ec-2c084e963efb





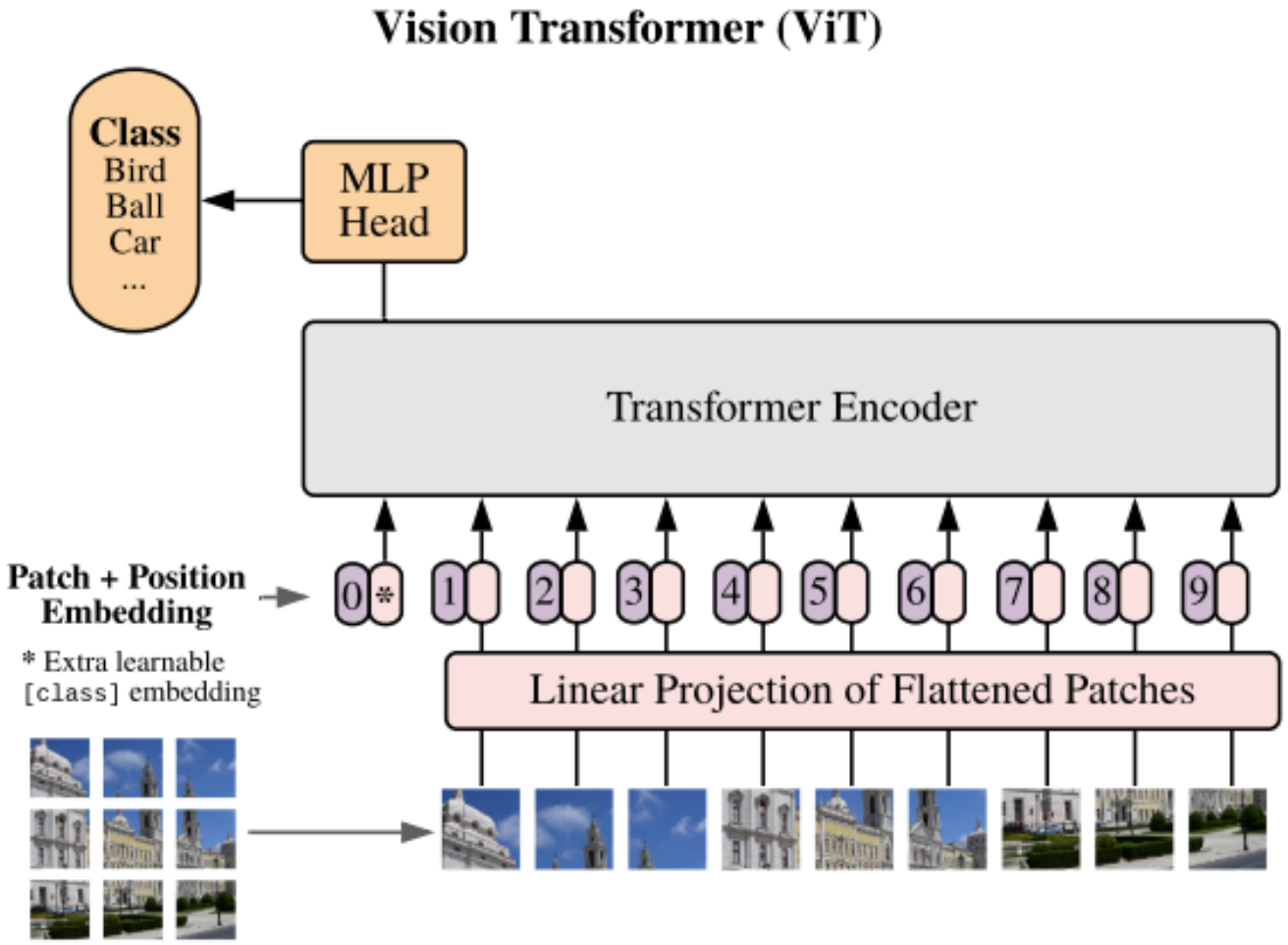
Generate latent

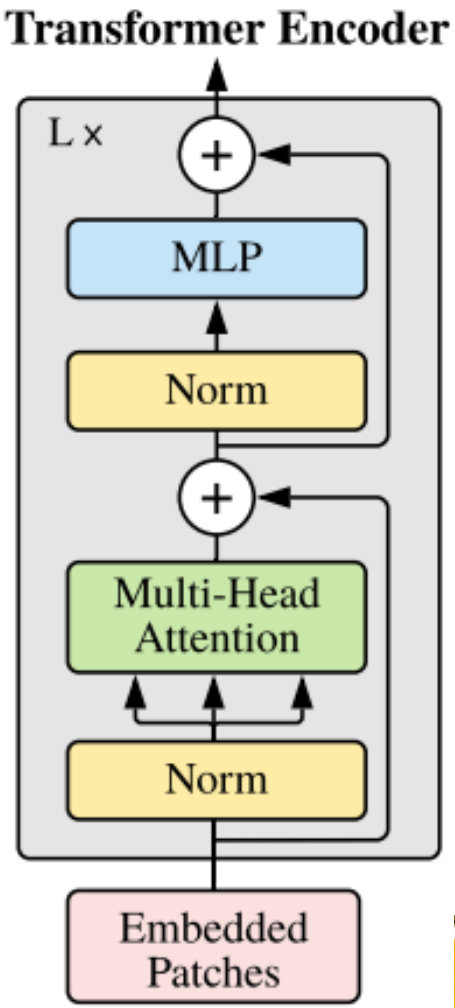


Latent array加權後
比較能取到七的位置

幫我數字7

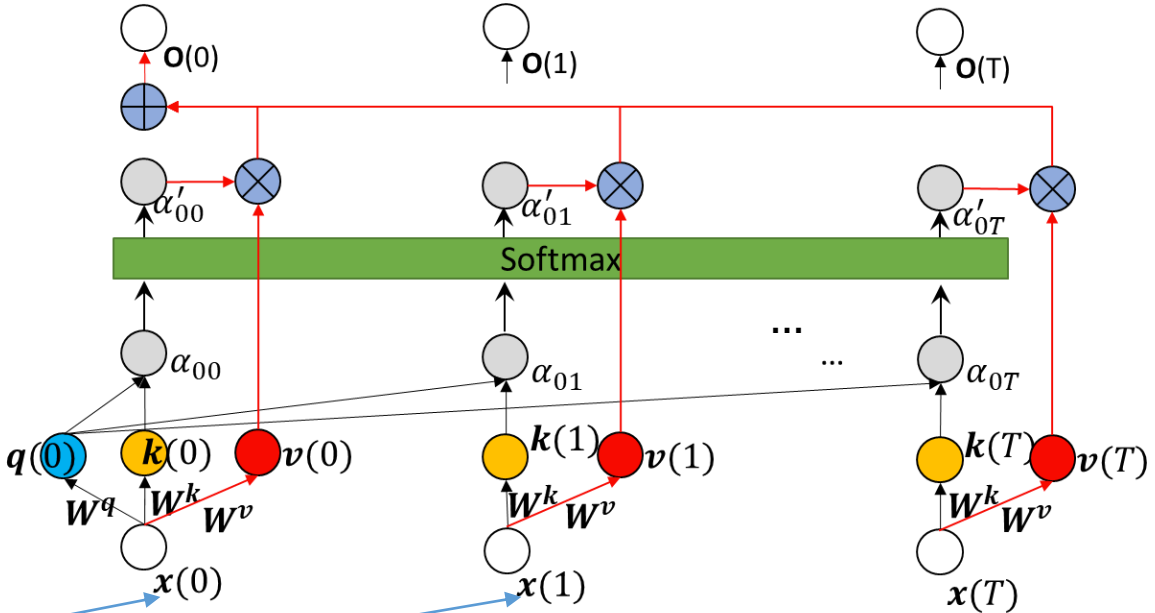
Vision Transformer(ViT)



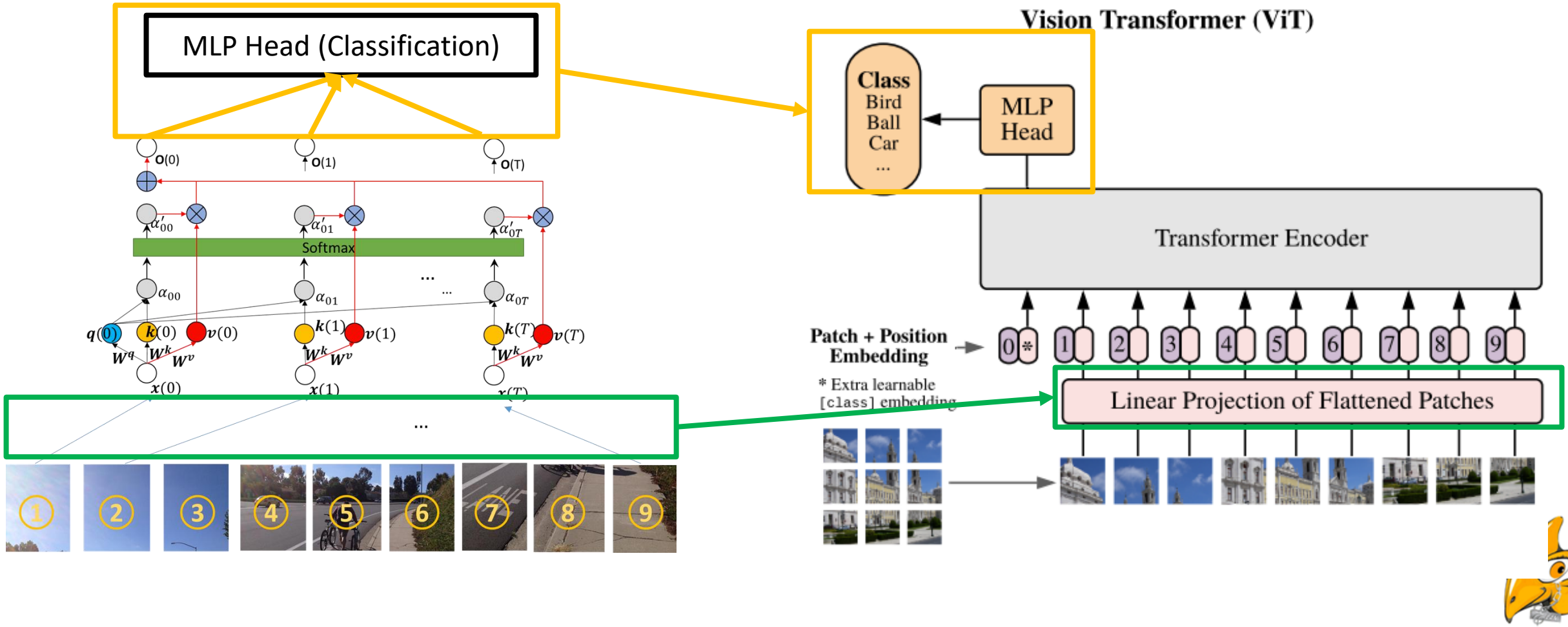


Vision Transformer(ViT)

Vision Transformer(ViT)



Vision Transformer(ViT)

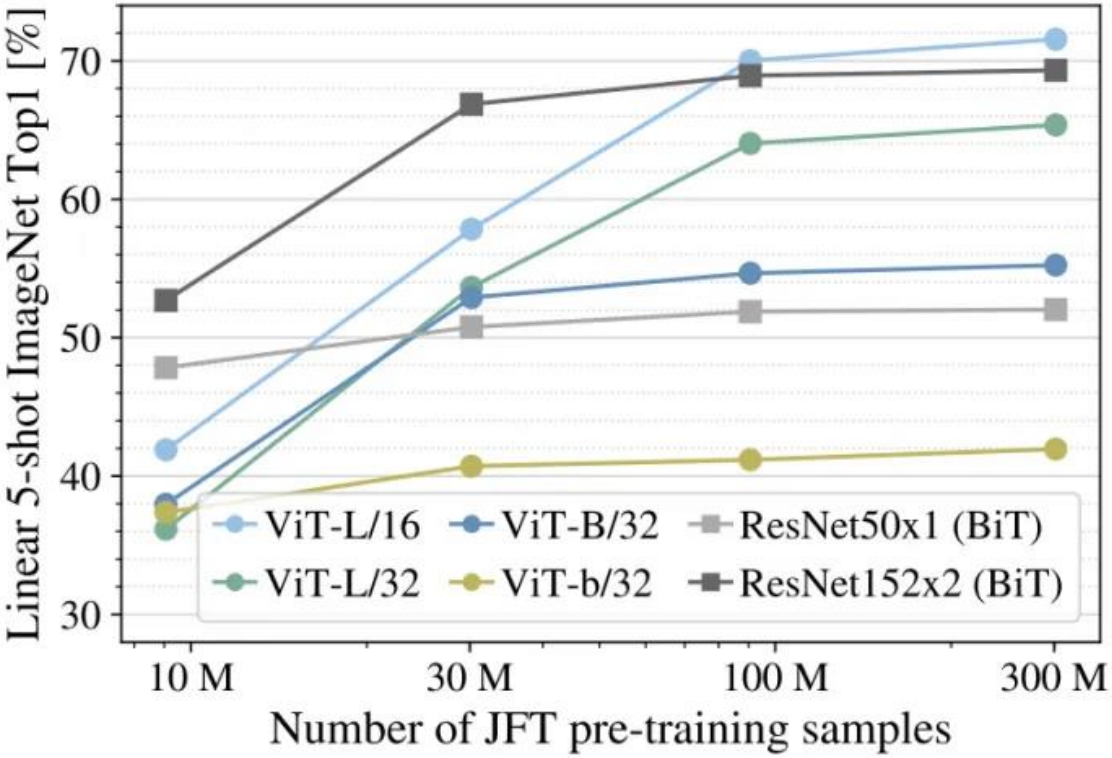


Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

Table 1: Details of Vision Transformer model variants.

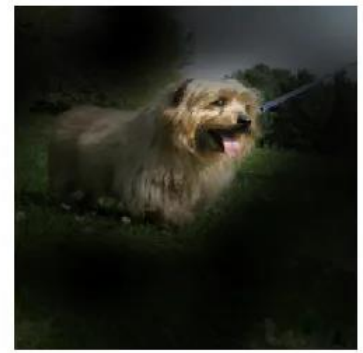
ViT-L/16表示的是large model使用的是16*16的patch

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21K (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k



Big Transform: BiT

Input Attention



CNN and Self-Attention

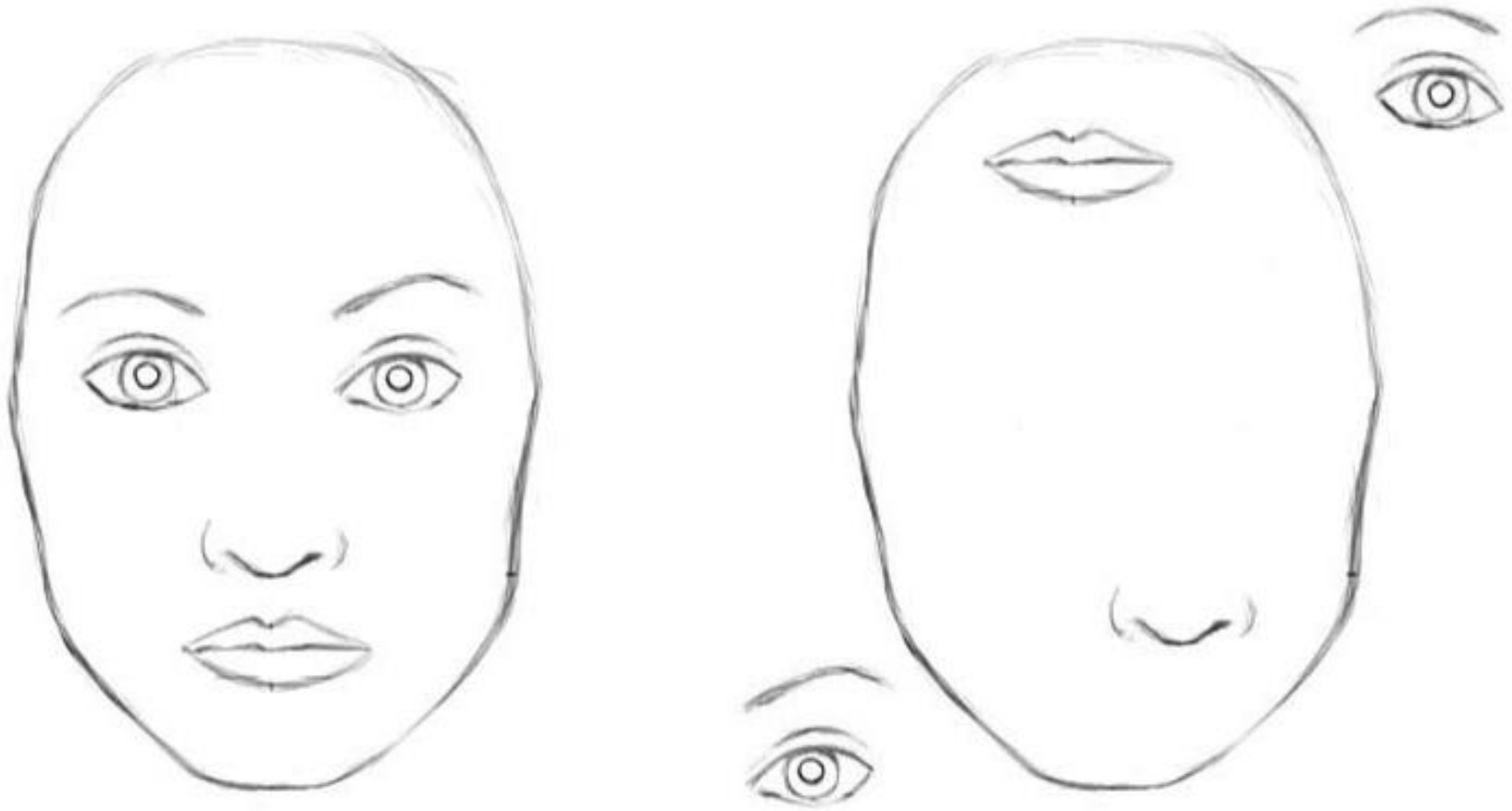
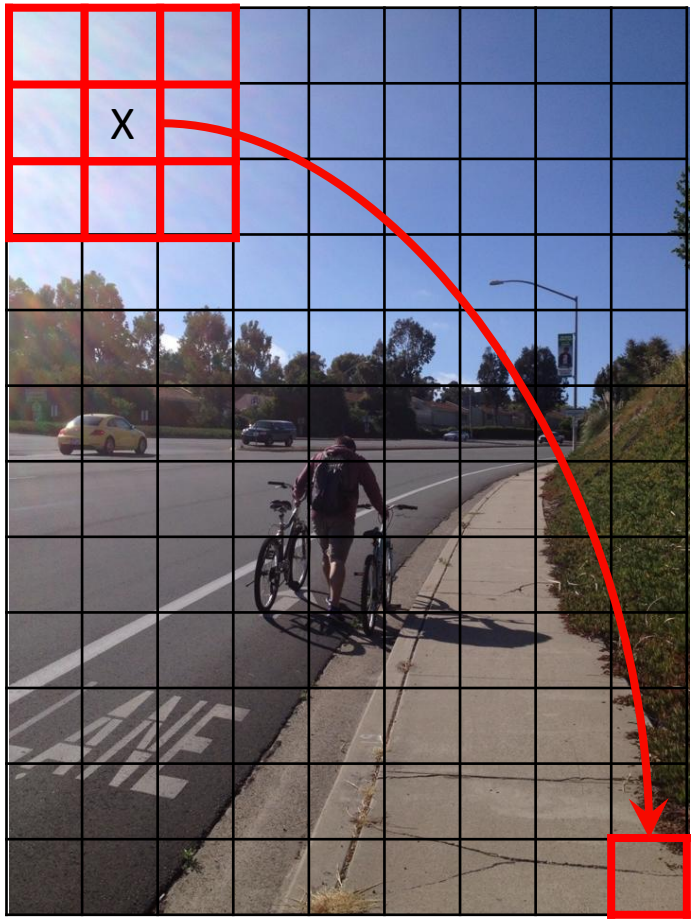


Fig 1: Both images are similar for a CNN as it does not consider the relative positioning of facial features

https://towardsdatascience.com/is-this-the-end-for-convolutional-neural-networks-6f944dccc2e9

CNN and Self-Attention

CNN
Receptive field



Self - Attention