MACHINE LEARNING

机器学习

Neural Networks

循环神经网络与注意力机制

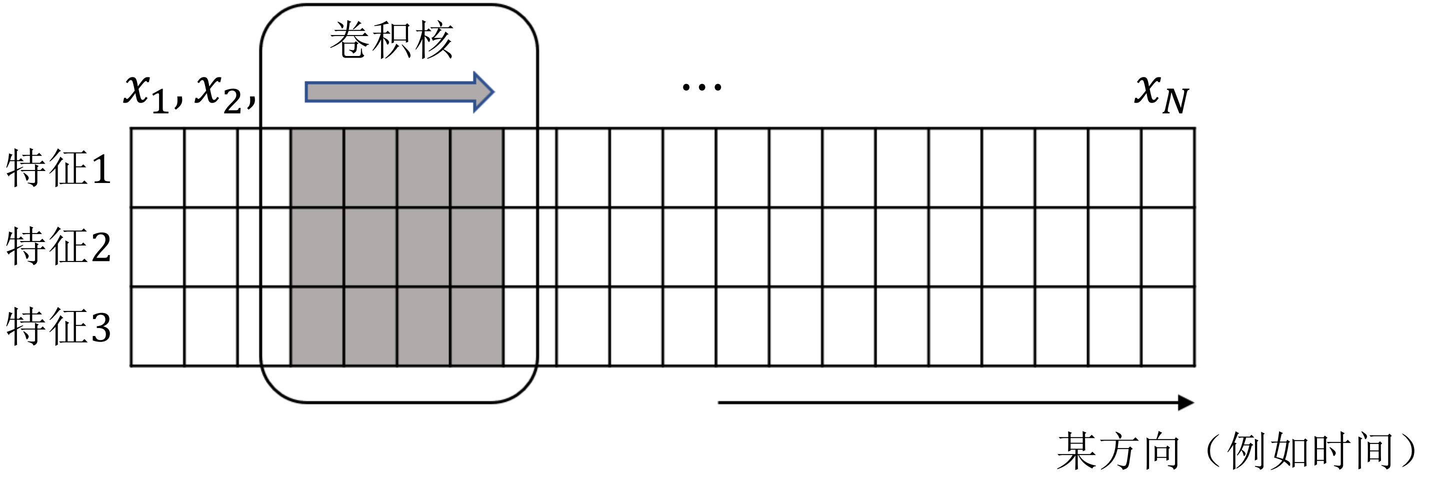
参考：
《神经网络与深度学习》

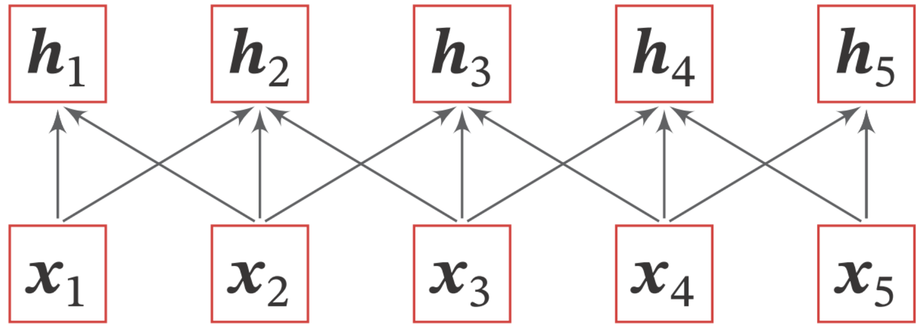
tu.wenting@mail.shufe.edu.cn

循环神经网络 *Recurrent Neural Network*

- 卷积层实现序列到序列

卷积层处理1D序列数据

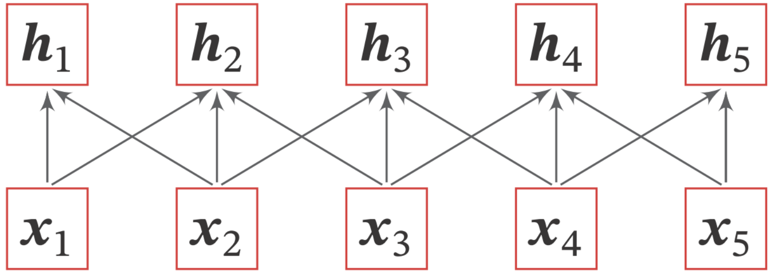




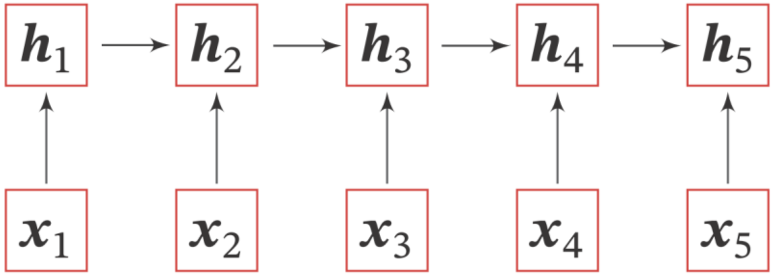
循环神经网络 *Recurrent Neural Network*

- 循环层实现序列到序列

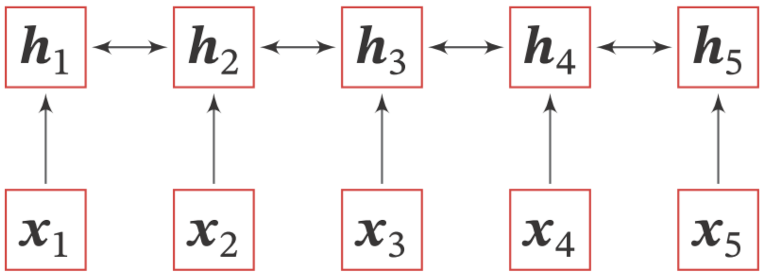
CNN



RNN



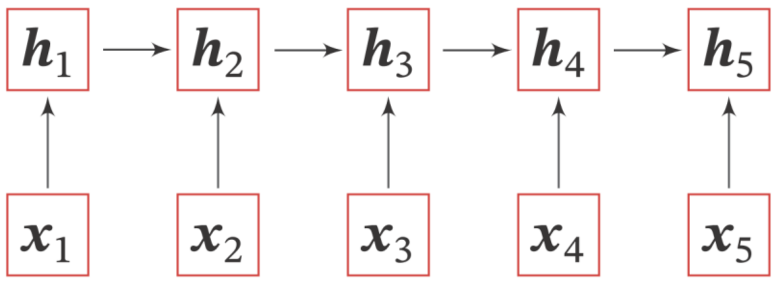
$$\boldsymbol{h}_t = f(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b})$$

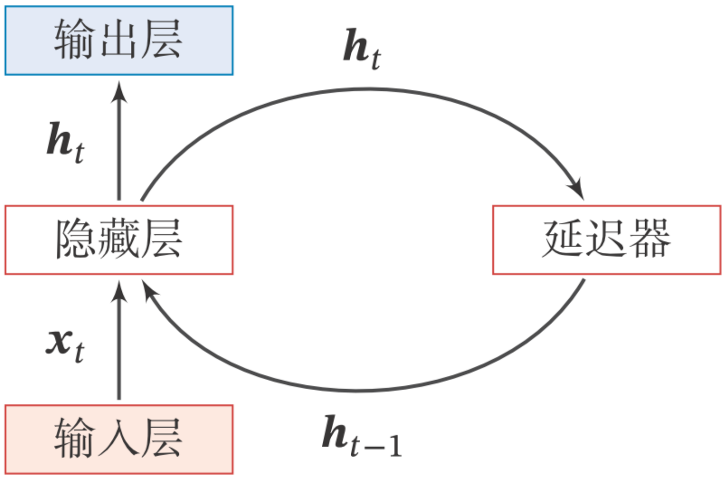


$$\boldsymbol{h}_t^{(1)} = f(\boldsymbol{U}^{(1)}\boldsymbol{h}_{t-1}^{(1)} + \boldsymbol{W}^{(1)}\boldsymbol{x}_t + \boldsymbol{b}^{(1)})$$
$$\boldsymbol{h}_t^{(2)} = f(\boldsymbol{U}^{(2)}\boldsymbol{h}_{t+1}^{(2)} + \boldsymbol{W}^{(2)}\boldsymbol{x}_t + \boldsymbol{b}^{(2)})$$
$$\boldsymbol{h}_t = \boldsymbol{h}_t^{(1)} \oplus \boldsymbol{h}_t^{(2)}$$

循环神经网络 *Recurrent Neural Network*

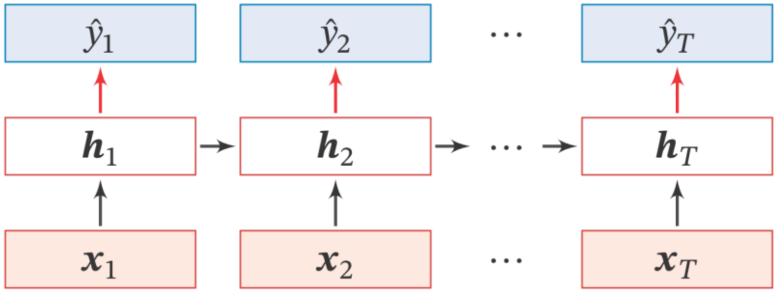
- 循环层实现序列到序列



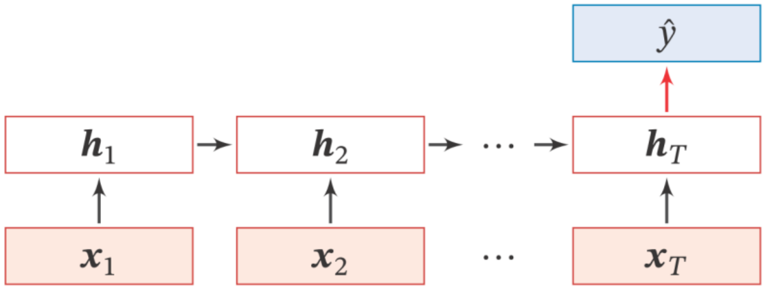


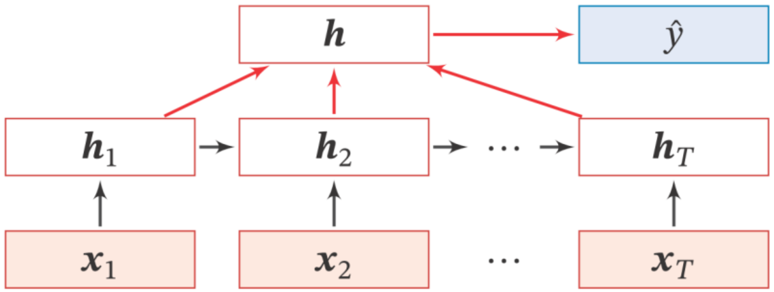
循环神经网络 *Recurrent Neural Network*

- 循环层上的输出



如果输出是一个同步的序列，可以
把每步输出建立在每步的隐层上



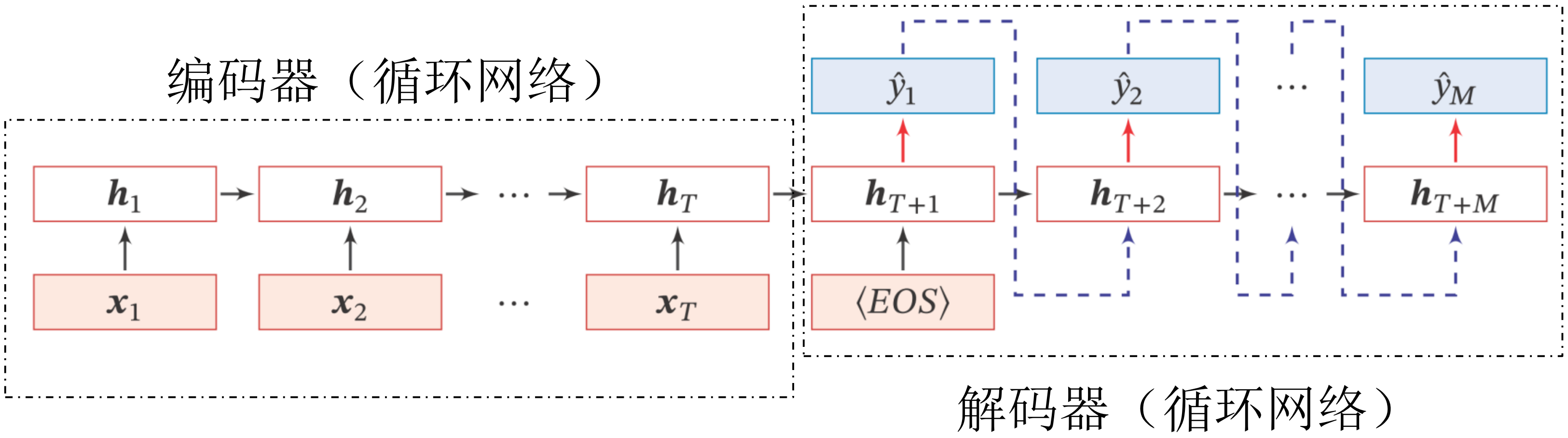


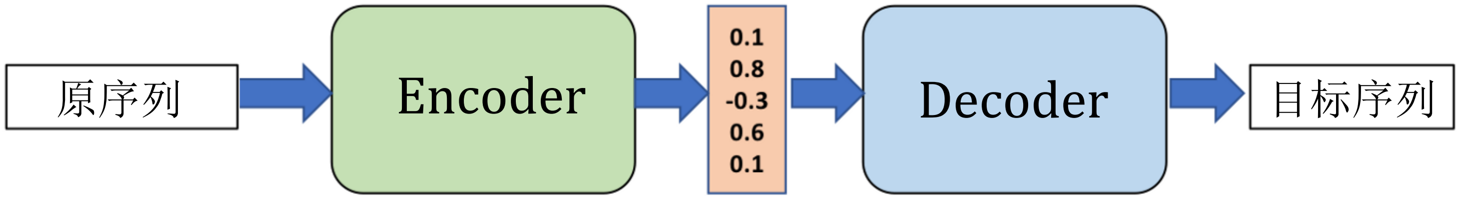
如果输出是一个标签，可以把输出建立最后一步隐层
或所有步的隐层上

循环神经网络 *Recurrent Neural Network*

- 异步的序列到序列

- 编码器-解码器（Encoder-Decoder）

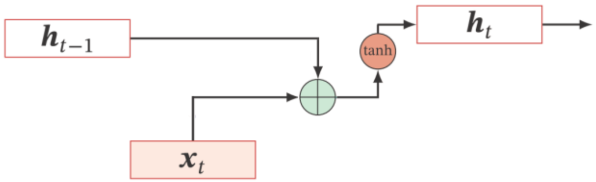




门控循环单元 *Gated Recurrent Unit*

- 门控制

为了改善循环神经网络的长程依赖问题，流行的方法是引入门控机制来控制信息的累积速度，包括有选择地遗忘之前累积的信息以及有选择地加入新的信息。

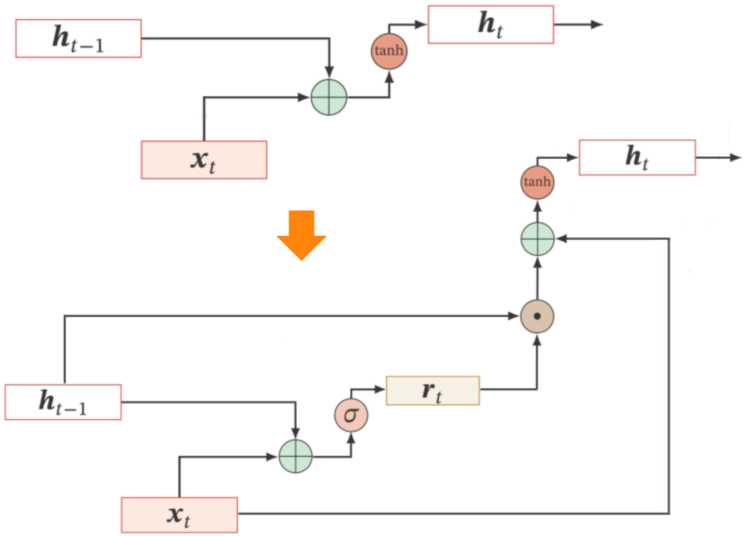


$$\boldsymbol{h}_t = f(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b})$$

门控循环单元 *Gated Recurrent Unit*

- 门控制

为了改善循环神经网络的长程依赖问题，流行的方法是引入门控机制来控制信息的累积速度，包括有选择地遗忘之前累积的信息以及有选择地加入新的信息。



$$h_t = f(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b})$$

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r\boldsymbol{x}_t + \boldsymbol{U}_r\boldsymbol{h}_{t-1} + \boldsymbol{b}_r)$$

$$\boldsymbol{h}_t = \tanh\left(\boldsymbol{W}_h\boldsymbol{x}_t + \boldsymbol{U}_h(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_h\right)$$

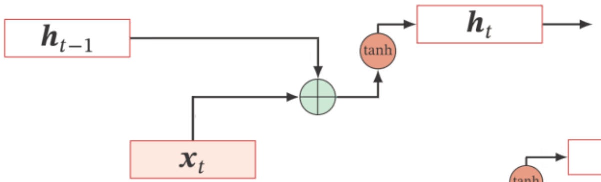
$\mathbf{r}_t$ 控制$\boldsymbol{h}_t$的计算是否/多大程度依赖上一时刻的状态$\boldsymbol{h}_{t-1}$

这允许我们转换当前信息的时候一定程度上遗忘历史信息

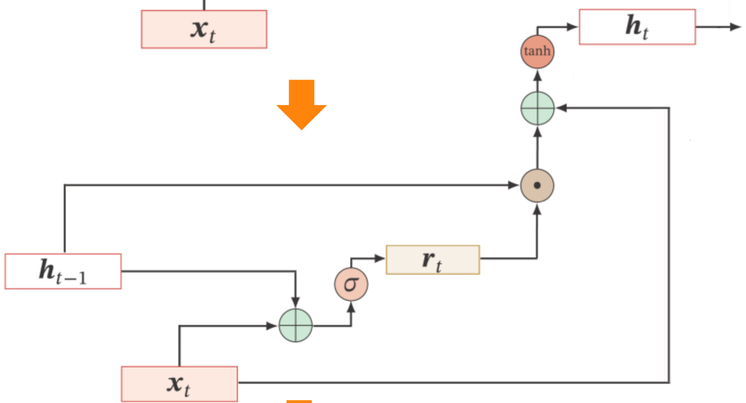
门控循环单元 *Gated Recurrent Unit*

- 门控制

为了改善循环神经网络的长程依赖问题，流行的方法是引入门控机制来控制信息的累积速度，包括有选择地遗忘之前累积的信息以及有选择地加入新的信息。



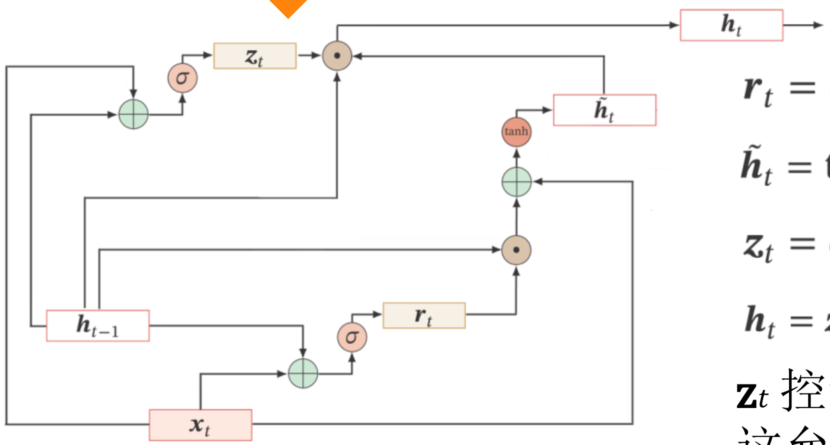
$$\boldsymbol{h}_t = f(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b})$$



$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r\boldsymbol{x}_t + \boldsymbol{U}_r\boldsymbol{h}_{t-1} + \boldsymbol{b}_r)$$

$$\boldsymbol{h}_t = \tanh\left(\boldsymbol{W}_h\boldsymbol{x}_t + \boldsymbol{U}_h(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_h\right)$$

$\mathbf{r}_t$ 控制$\boldsymbol{h}_t$的计算是否/多大程度依赖上一时刻的状态$\boldsymbol{h}_{t-1}$
这允许我们转换当前信息的时候一定程度上遗忘历史信息



$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r\boldsymbol{x}_t + \boldsymbol{U}_r\boldsymbol{h}_{t-1} + \boldsymbol{b}_r)$$

$$\tilde{\boldsymbol{h}}_t = \tanh\left(\boldsymbol{W}_h\boldsymbol{x}_t + \boldsymbol{U}_h(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_h\right)$$

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_z\boldsymbol{x}_t + \boldsymbol{U}_z\boldsymbol{h}_{t-1} + \boldsymbol{b}_z)$$

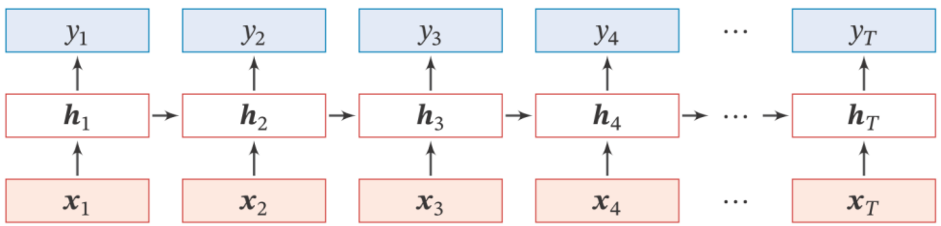
$$\boldsymbol{h}_t = \boldsymbol{z}_t \odot \boldsymbol{h}_{t-1} + (1 - \boldsymbol{z}_t) \odot \tilde{\boldsymbol{h}}_t$$

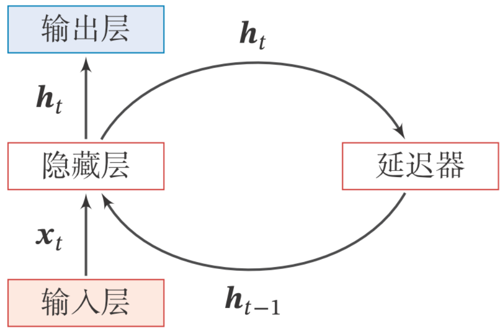
Gated Recurrent Unit， GRU

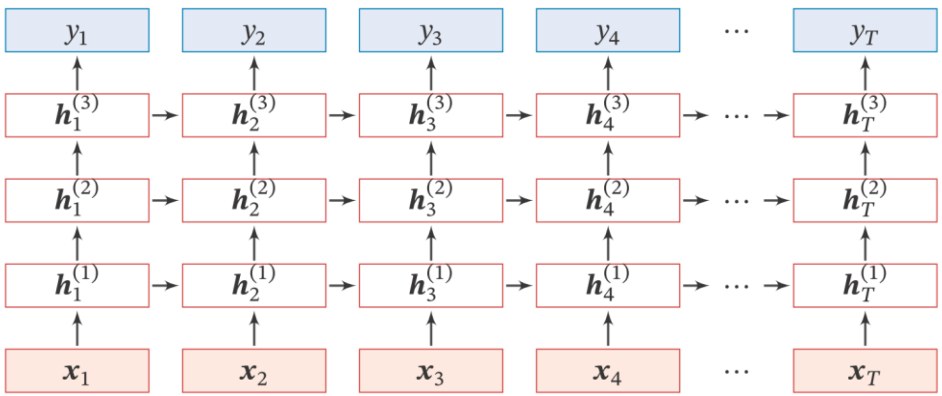
$\mathbf{z}_t$ 控制$\boldsymbol{h}_t$的计算是如何融合当前信息与过去信息的
这允许我们在输出信息的时候一定程度上遗忘当前

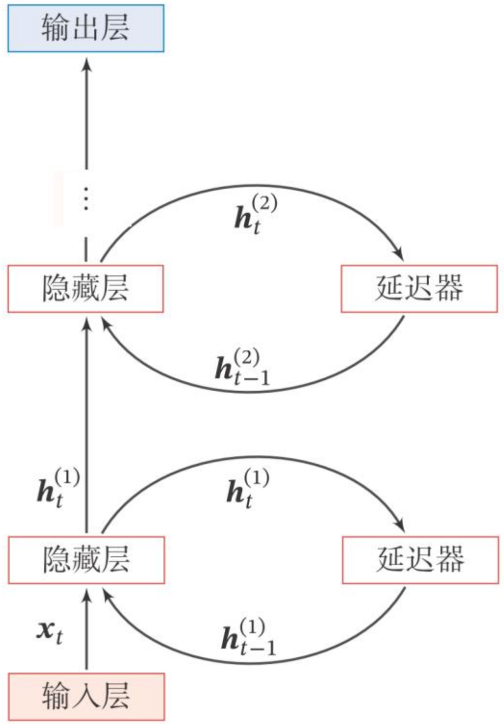
其他变体

- 堆叠循环神经网络





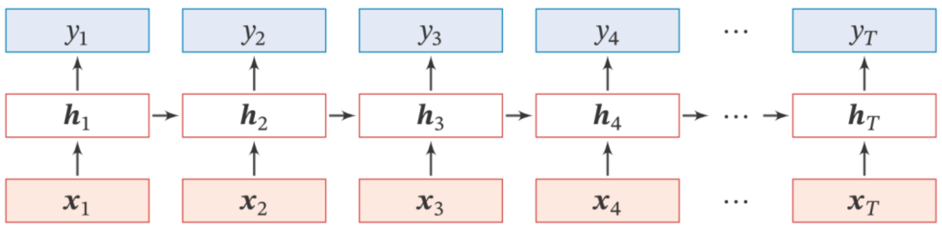


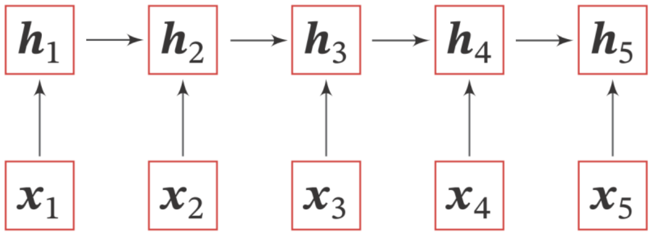


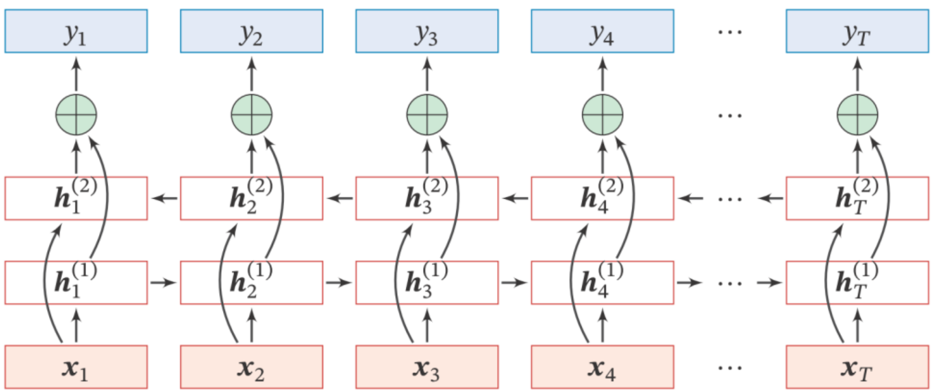
$$h_t^{(l)} = f(U^{(l)} h_{t-1}^{(l)} + W^{(l)} h_t^{(l-1)} + b^{(l)})$$

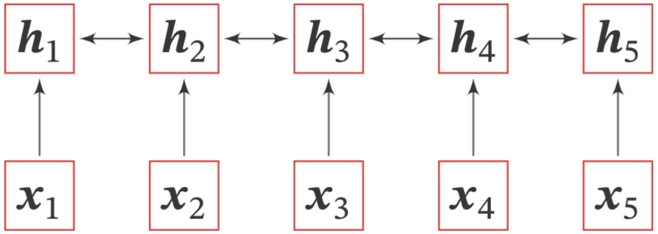
其他变体

- 双向循环神经网络









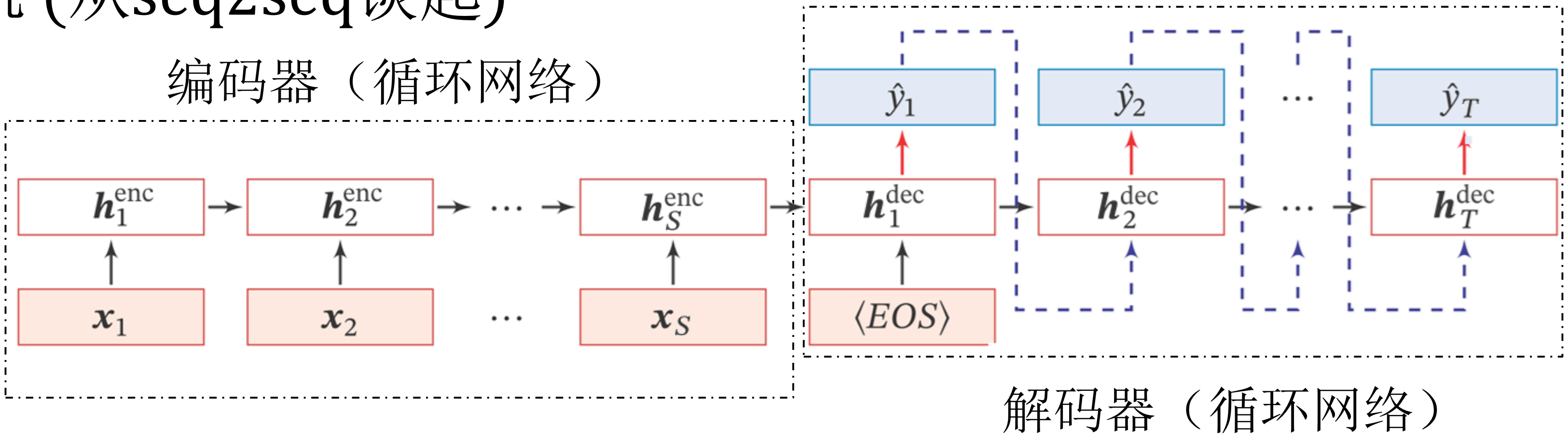
$$\boldsymbol{h}_t^{(1)} = f(\boldsymbol{U}^{(1)}\boldsymbol{h}_{t-1}^{(1)} + \boldsymbol{W}^{(1)}\boldsymbol{x}_t + \boldsymbol{b}^{(1)})$$

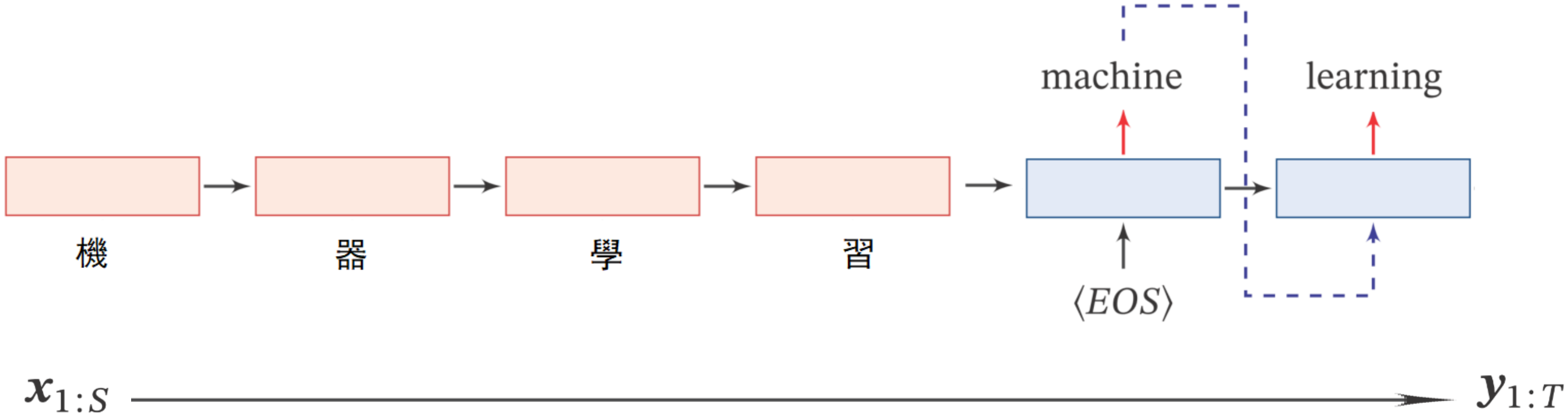
$$\boldsymbol{h}_t^{(2)} = f(\boldsymbol{U}^{(2)}\boldsymbol{h}_{t+1}^{(2)} + \boldsymbol{W}^{(2)}\boldsymbol{x}_t + \boldsymbol{b}^{(2)})$$

$$\boldsymbol{h}_t = \boldsymbol{h}_t^{(1)} \oplus \boldsymbol{h}_t^{(2)}$$

注意力机制

- 动机 (从seq2seq谈起)





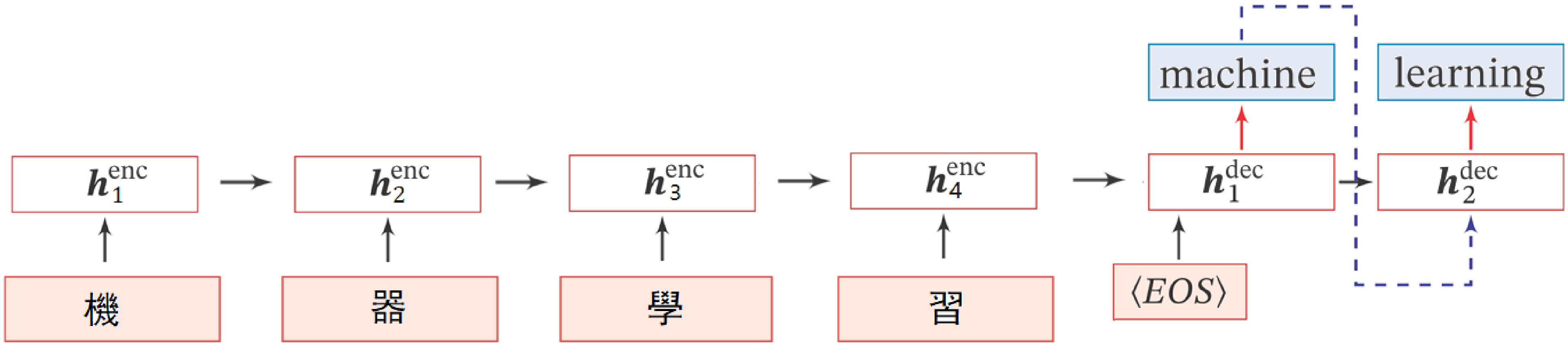
$$h_t^{\text{enc}} = f_{\text{enc}}(h_{t-1}^{\text{enc}}, e_{x_t}, \theta_{\text{enc}})$$
$$\forall t \in [1 : S]$$

$$u = h_S^{\text{enc}}$$

$$h_0^{\text{dec}} = u$$
$$h_t^{\text{dec}} = f_{\text{dec}}(h_{t-1}^{\text{dec}}, e_{y_{t-1}}, \theta_{\text{dec}})$$
$$o_t = g(h_t^{\text{dec}}, \theta_o)$$

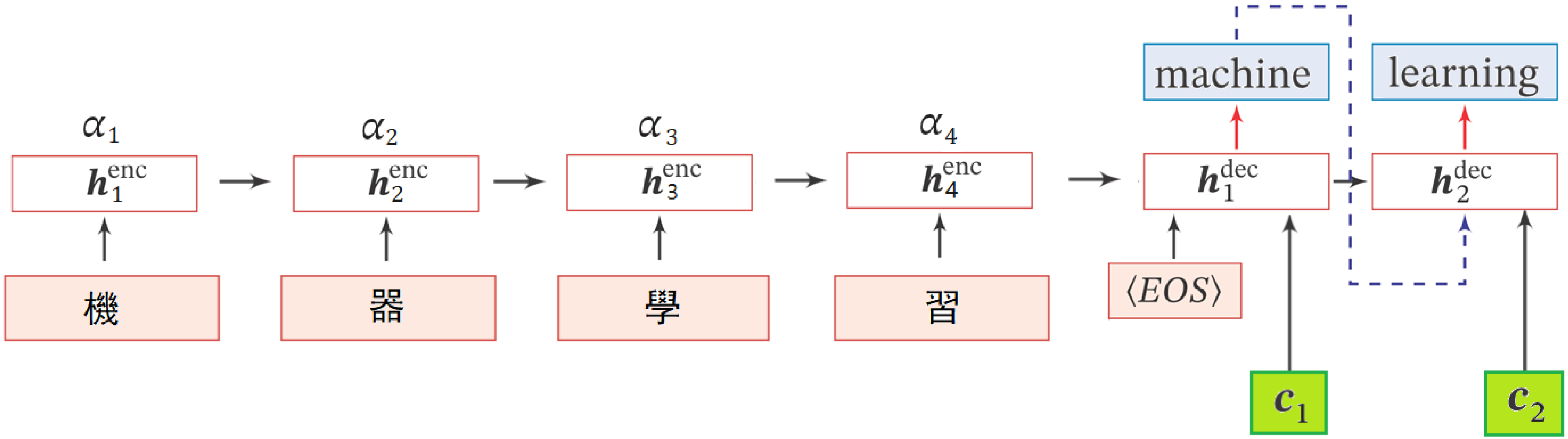
注意力机制

- 动机 (从seq2seq谈起)



注意力机制

- 动机 (从seq2seq谈起)

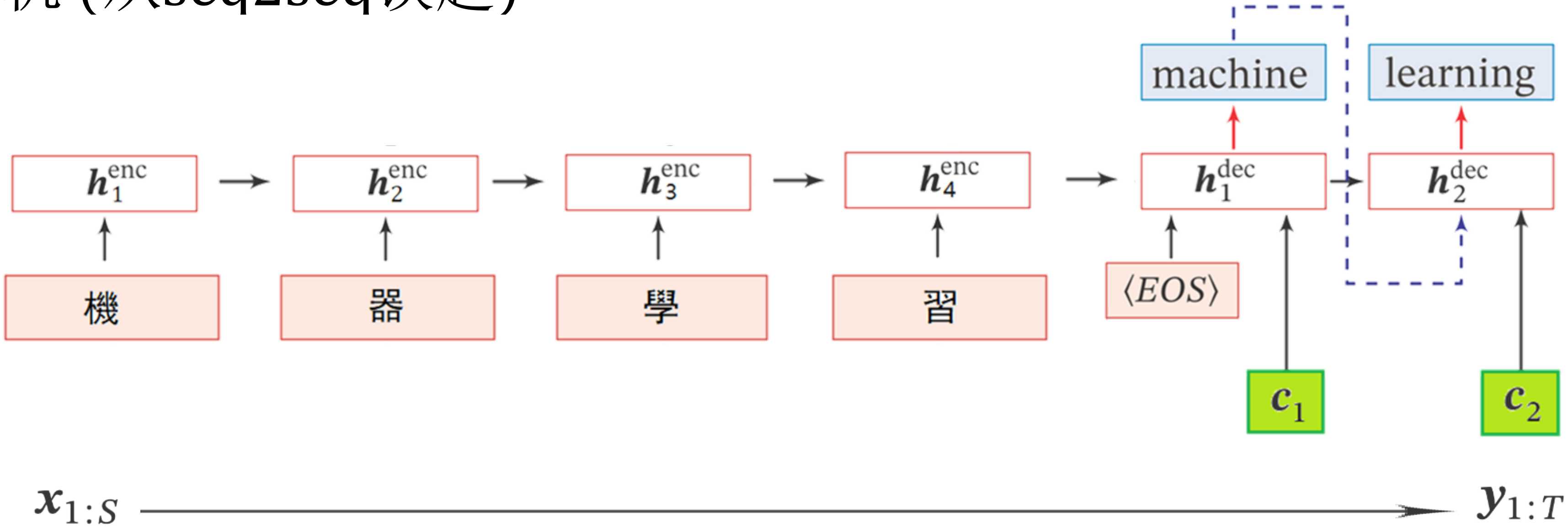


$$c_1 = 0.5\, h_1^{enc} + 0.5\, h_2^{enc} + 0.0\, h_3^{enc} + 0.0\, h_4^{enc}$$

$$c_2 = 0.0\, h_1^{enc} + 0.0\, h_2^{enc} + 0.5\, h_3^{enc} + 0.5\, h_4^{enc}$$

注意力机制

- 动机 (从seq2seq谈起)



$$\boldsymbol{h}_t^{\text{enc}} = f_{\text{enc}}(\boldsymbol{h}_{t-1}^{\text{enc}}, \boldsymbol{e}_{x_t}, \theta_{\text{enc}})$$

$$\forall t \in [1 : S]$$

$$\boldsymbol{H}^{\text{enc}} = [\boldsymbol{h}_1^{\text{enc}}, \cdots, \boldsymbol{h}_S^{\text{enc}}]$$

$$\boldsymbol{u} = \boldsymbol{h}_S^{\text{enc}}$$

$$\boldsymbol{h}_0^{\text{dec}} = \boldsymbol{u}$$

$$\boldsymbol{c}_t = \text{att}(\boldsymbol{H}^{\text{enc}}, \boldsymbol{h}_{t-1}^{\text{dec}}) = \sum_{i=1}^{S} \alpha_i \boldsymbol{h}_i^{\text{enc}}$$

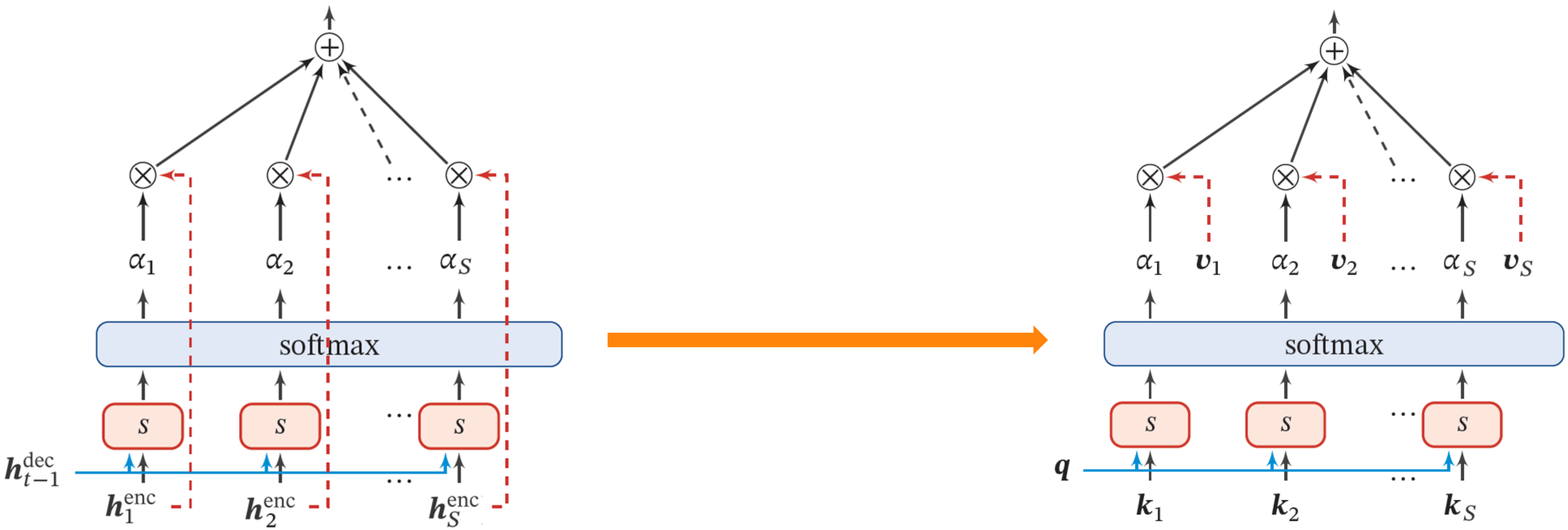
$$= \sum_{i=1}^{S} \text{softmax}\Big(s(\boldsymbol{h}_i^{\text{enc}}, \boldsymbol{h}_{t-1}^{\text{dec}})\Big)\boldsymbol{h}_i^{\text{enc}}$$

$$\boldsymbol{h}_t^{\text{dec}} = f_{\text{dec}}(\boldsymbol{h}_{t-1}^{\text{dec}}, [\boldsymbol{e}_{y_{t-1}}; \boldsymbol{c}_t], \theta_{\text{dec}})$$

$$\boldsymbol{o}_t = g(\boldsymbol{h}_t^{\text{dec}}, \theta_o)$$

注意力机制

- 键值对注意力



$$\sum_{i=1}^{S}\alpha_i \boldsymbol{h}_i^{\mathrm{enc}} = \sum_{i=1}^{S}\mathrm{softmax}\Big(s(\boldsymbol{h}_i^{\mathrm{enc}}, \boldsymbol{h}_{t-1}^{\mathrm{dec}})\Big)\boldsymbol{h}_i^{\mathrm{enc}}$$

Query

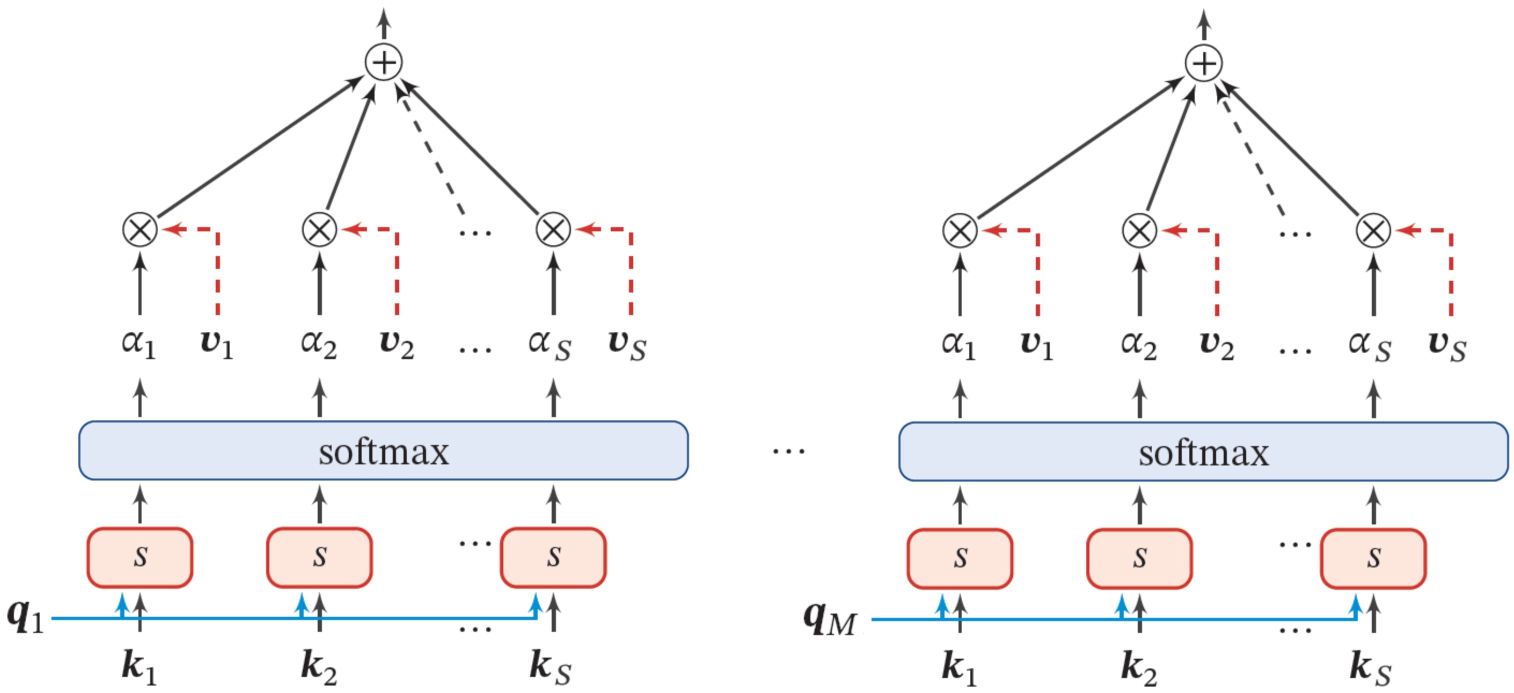
Key, Value

$$\mathrm{att}\Big((\boldsymbol{K},\boldsymbol{V}),\boldsymbol{q}\Big) = \sum_{i=1}^{S}\alpha_i \boldsymbol{v}_i = \sum_{i=1}^{S}\frac{\exp\Big(s(\boldsymbol{k}_i,\boldsymbol{q})\Big)}{\sum_j \exp\Big(s(\boldsymbol{k}_j,\boldsymbol{q})\Big)}\boldsymbol{v}_i$$

当 $\boldsymbol{K}=\boldsymbol{V}$ 时，键值对模式就等价于普通的注意力机制

注意力机制

- 多头注意力

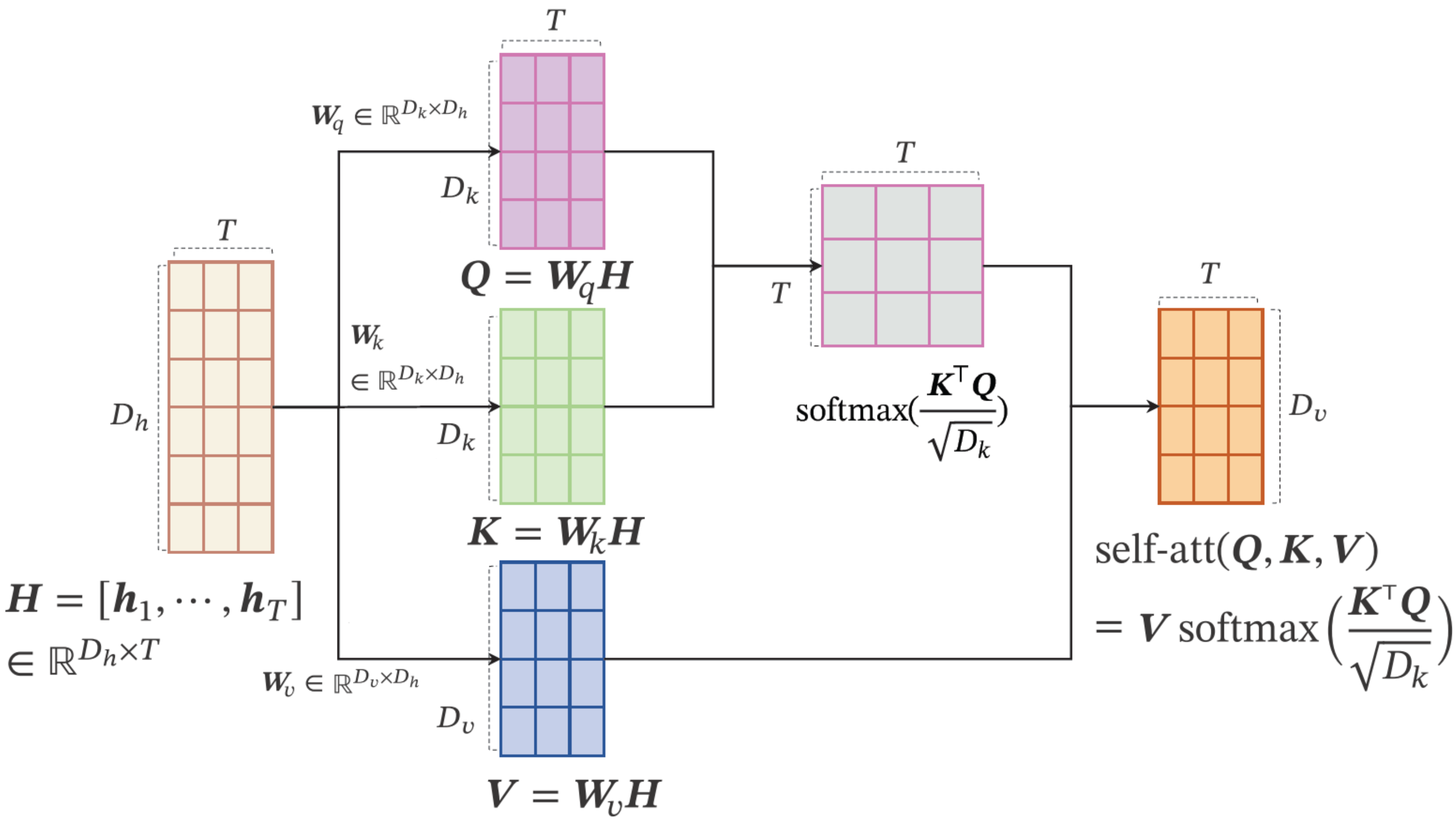


利用多个查询 $\boldsymbol{Q} = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_M]$

$$\text{att}\Big((\boldsymbol{K},\boldsymbol{V}),\boldsymbol{Q}\Big) = \text{att}\Big((\boldsymbol{K},\boldsymbol{V}),\boldsymbol{q}_1\Big) \oplus \cdots \oplus \text{att}\Big((\boldsymbol{K},\boldsymbol{V}),\boldsymbol{q}_M\Big)$$

注意力机制

- 自注意力



T
$W_q \in \mathbb{R}^{D_k \times D_h}$
D_k
$Q = W_q H$
T
T
T
D_h
W_k
$\in \mathbb{R}^{D_k \times D_h}$
D_k
softmax$(\frac{K^\top Q}{\sqrt{D_k}})$
T
D_v
$K = W_k H$
self-att(Q, K, V)
$= V \,\mathrm{softmax}\left(\frac{K^\top Q}{\sqrt{D_k}}\right)$
$H = [h_1, \cdots, h_T]$
$\in \mathbb{R}^{D_h \times T}$
$W_v \in \mathbb{R}^{D_v \times D_h}$
D_v
$V = W_v H$