循环神经网络

Recurrent Neural Network

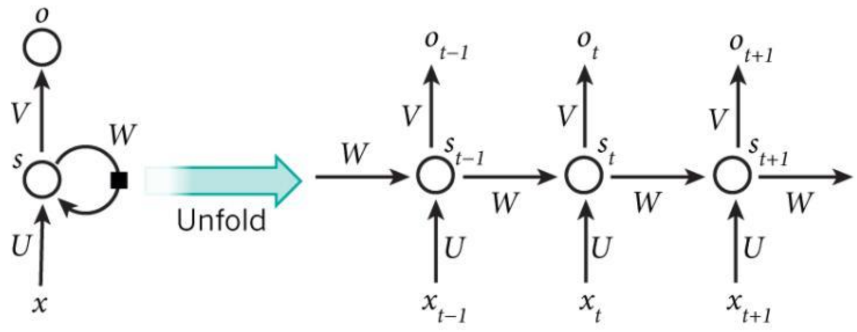
Recurrent Neural Network

- 起源：
  1982 John Joseph Hopfield提出带有反馈结构的Hopfield网络
  Neural networks and physical systems with emergent collective computational abilities
- 发展：
  1986 Michael Jordan提出循环结构的
  Serial order: A parallel distributed processing approach

  1990 Jeffrey Elman提出现代意义上的Recurrent Neural Network
  Finding structure in time

- 壮大：
  1997 Hochreiter & Schmidhuber提出LSTM
  LSTM: long short-term memory for vanishing gradients problem

Recurrent Neural Network



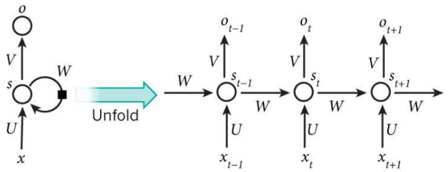
$$x \in \mathbb{R}(m)$$
$$U \in \mathbb{R}(n, m)$$
$$s_{Ux} = U \cdot x + b_s$$
$$\rightarrow s_{Ux} \in \mathbb{R}(n)$$

$$W \in \mathbb{R}(n, n)$$
$$s_{Ws} = W \cdot s_{t-1} + b_s$$
$$\rightarrow s_{Ws} \in \mathbb{R}(n)$$

$$o_t = g(Vs_t) \tag{1}$$
$$s_t = f(Ux_t + Ws_{t-1}) \tag{2}$$

$$\begin{aligned} \longrightarrow \quad o_t &= g(Vs_t) \\ &= g(Vf(Ux_t + Ws_{t-1})) \\ &= g(Vf(Ux_t + Wf(Ux_{t-1} + Ws_{t-2}))) \\ &= g(Vf(Ux_t + Wf(Ux_{t-1} + Wf(Ux_{t-2} + Ws_{t-3})))) \\ &= g(Vf(Ux_t + Wf(Ux_{t-1} + Wf(Ux_{t-2} + Wf(Ux_{t-3} + \cdots))))) \end{aligned} \tag{3}$$

Recurrent Neural Network



$$o_t = g(Vs_t) \quad (1)$$
$$s_t = f(Ux_t + Ws_{t-1}) \quad (2)$$

$$\begin{aligned} o_t &= g(Vs_t) \\ &= g(Vf(Ux_t + Ws_{t-1})) \\ &= g(Vf(Ux_t + Wf(Ux_{t-1} + Ws_{t-2}))) \\ &= g(Vf(Ux_t + Wf(Ux_{t-1} + Wf(Ux_{t-2} + Ws_{t-3})))) \\ &= g(Vf(Ux_t + Wf(Ux_{t-1} + Wf(Ux_{t-2} + Wf(Ux_{t-3} + \cdots))))) \end{aligned} \quad (3)$$

加入偏置

$$o_t = g(Vs_t + b_o) \quad (1)$$
$$s_t = f(Ux_t + Ws_{t-1} + b_s) \quad (2)$$

$$f(z) = tanh(z)$$
$$g(z) = softmax(z)$$

$$U = \begin{bmatrix} 0.10 & 0.20 & 0.30 \\ 0.40 & 0.50 & 0.60 \end{bmatrix}, \quad W = \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix}, V = \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix}, b_s = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} b_o = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix}$$

$$x_1 = \begin{bmatrix} 1.00 \\ 2.00 \\ 3.00 \end{bmatrix}, x_2 = \begin{bmatrix} 2.00 \\ 3.00 \\ 4.00 \end{bmatrix} x_3 = \begin{bmatrix} 3.00 \\ 4.00 \\ 5.00 \end{bmatrix}$$

1,2,3,4,5

$$GroundTruth_1 = \begin{bmatrix} 1.00 \\ 0.00 \\ 0.00 \end{bmatrix} GroundTruth_2 = \begin{bmatrix} 0.00 \\ 0.00 \\ 1.00 \end{bmatrix} GroundTruth_3 = \begin{bmatrix} 0.00 \\ 1.00 \\ 0.00 \end{bmatrix}$$

Recurrent Neural Network

$$s_1 = tanh(U \cdot x_1 + W \cdot s_0 + b_s)$$
$$= tanh(\begin{bmatrix} 0.10 & 0.20 & 0.30 \\ 0.40 & 0.50 & 0.60 \end{bmatrix} \cdot \begin{bmatrix} 1.00 \\ 2.00 \\ 3.00 \end{bmatrix} + \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix} \cdot \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} + \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix})$$
$$= tanh(\begin{bmatrix} 1.40 \\ 3.20 \end{bmatrix})$$
$$= \begin{bmatrix} 0.88535 \\ 0.99668 \end{bmatrix}$$

$$o_1 = softmax(V \cdot s_1 + b_o)$$
$$= softmax(\begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix} \cdot \begin{bmatrix} 0.88535 \\ 0.99668 \end{bmatrix} + \begin{bmatrix} 0.00 \\ 0.00 \\ 0.00 \end{bmatrix})$$
$$= softmax(\begin{bmatrix} 0.29 \\ 0.66 \\ 1.04 \end{bmatrix})$$
$$= \begin{bmatrix} 0.22 \\ 0.32 \\ 0.46 \end{bmatrix}$$

$$s_2 = tanh(U \cdot x_2 + W \cdot s_1 + b_s)$$
$$= tanh(\begin{bmatrix} 0.10 & 0.20 & 0.30 \\ 0.40 & 0.50 & 0.60 \end{bmatrix} \cdot \begin{bmatrix} 2.00 \\ 3.00 \\ 4.00 \end{bmatrix} + \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix} \cdot \begin{bmatrix} 0.89 \\ 1.00 \end{bmatrix} + \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix})$$
$$= tanh(\begin{bmatrix} 2.29 \\ 5.36 \end{bmatrix})$$
$$= \begin{bmatrix} 0.97961 \\ 0.99996 \end{bmatrix}$$

$$o_2 = softmax(V \cdot s_2 + b_o)$$
$$= softmax(\begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix} \cdot \begin{bmatrix} 0.97961 \\ 0.99996 \end{bmatrix} + \begin{bmatrix} 0.00 \\ 0.00 \\ 0.00 \end{bmatrix})$$
$$= softmax(\begin{bmatrix} 0.30 \\ 0.69 \\ 1.09 \end{bmatrix})$$
$$= \begin{bmatrix} 0.21 \\ 0.32 \\ 0.47 \end{bmatrix}$$

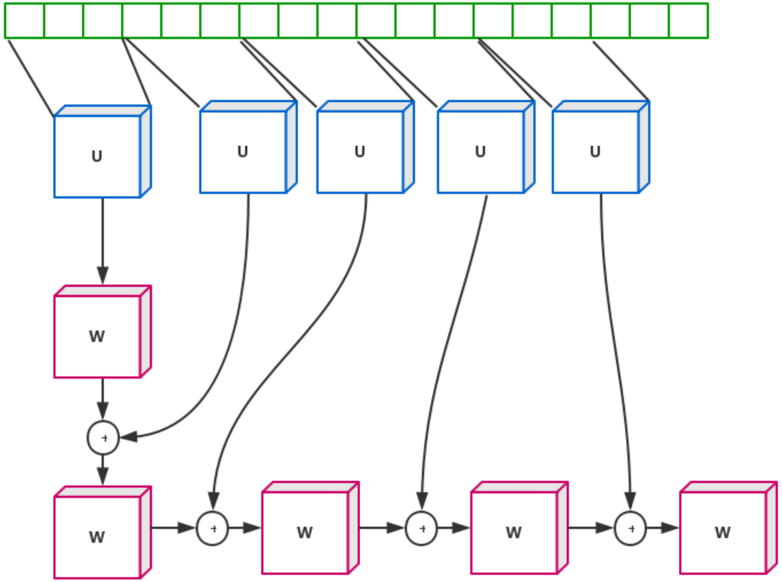
$$s_3 = tanh(U \cdot x_3 + W \cdot s_2 + b_s)$$
$$= tanh(\begin{bmatrix} 0.10 & 0.20 & 0.30 \\ 0.40 & 0.50 & 0.60 \end{bmatrix} \cdot \begin{bmatrix} 3.00 \\ 4.00 \\ 5.00 \end{bmatrix} + \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix} \cdot \begin{bmatrix} 0.98 \\ 1.00 \end{bmatrix} + \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix})$$
$$= tanh(\begin{bmatrix} 2.90 \\ 6.89 \end{bmatrix})$$
$$= \begin{bmatrix} 0.99394 \\ 1.00000 \end{bmatrix}$$

$$o_3 = softmax(V \cdot s_3 + b_o)$$
$$= softmax(\begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix} \cdot \begin{bmatrix} 0.99394 \\ 1.00000 \end{bmatrix} + \begin{bmatrix} 0.00 \\ 0.00 \\ 0.00 \end{bmatrix})$$
$$= softmax(\begin{bmatrix} 0.30 \\ 0.70 \\ 1.10 \end{bmatrix})$$
$$= \begin{bmatrix} 0.21 \\ 0.32 \\ 0.47 \end{bmatrix}$$

$$U = \begin{bmatrix} 0.10 & 0.20 & 0.30 \\ 0.40 & 0.50 & 0.60 \end{bmatrix}, \quad W = \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix}, V = \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix}, b_s = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} b_o = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix}$$
$$x_1 = \begin{bmatrix} 1.00 \\ 2.00 \\ 3.00 \end{bmatrix}, x_2 = \begin{bmatrix} 2.00 \\ 3.00 \\ 4.00 \end{bmatrix} x_3 = \begin{bmatrix} 3.00 \\ 4.00 \\ 5.00 \end{bmatrix}$$
$$GroundTruth_1 = \begin{bmatrix} 1.00 \\ 0.00 \\ 0.00 \end{bmatrix} GroundTruth_2 = \begin{bmatrix} 0.00 \\ 0.00 \\ 1.00 \end{bmatrix} GroundTruth_3 = \begin{bmatrix} 0.00 \\ 1.00 \\ 0.00 \end{bmatrix}$$

Recurrent Neural Network

- 在时间维度上：
  - 在输入上滑动
  - 对state进行深度计算



Recurrent Neural Network

- 沿时间反向传播（Back Propagation Through Time, BPTT）：
  - Loss为各个时间点的Loss之和
  - 权重的梯度同样为各个时间点的梯度之和
  - 前馈传播中的权重包括了：
    U：链接x_t（输入）和s_t（状态）
    W：链接s_{t-1}和s_t
    V：链接s_t和o_t（输出）
    所以我们针对这三个权重来考虑
  - $\delta_t^{(L)}$ 仍然定义为梯度参数，即 $\dfrac{\partial Cost}{\partial logits_t^{(L)}}$
  - 回忆一下普通神经网络时代的定义：

$$\delta^{(L)} = \nabla_y Cost \odot \sigma'(logit^{(L)}) \qquad (BP1)$$

$$\delta^{(l)} = ((w^{(l+1)})^T \delta^{(l+1)}) \odot \sigma'(logit^{(l)}) \qquad (BP2)$$

，且

$$o_t = g(Vs_t + b_o) \qquad (1)$$

$$s_t = f(Ux_t + Ws_{t-1} + b_s) \qquad (2)$$

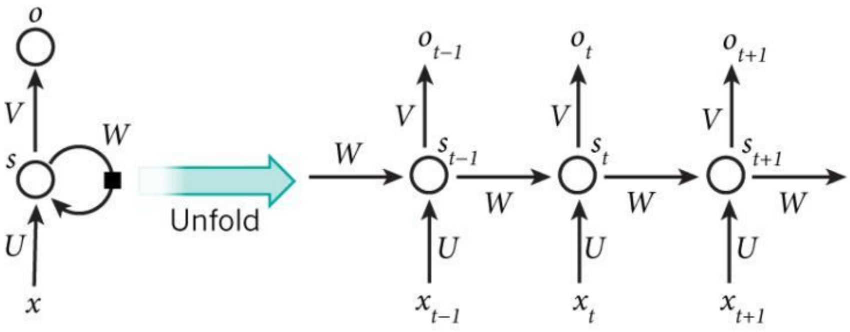
循环神经网络的反向传播

- 反向传播：
  - Loss为各个时间点的Loss之和
  - 权重的梯度同样为各个时间点的梯度之和
  - 前馈传播中的权重包括了：
    U：链接x_t（输入）和s_t（状态）
    W：链接s_{t-1}和s_t
    V：链接s_t和o_t（输出）
    所以我们针对这三个权重来考虑
  - $\delta_t^{(L)}$ 仍然定义为梯度参数，即 $\frac{\partial Cost}{\partial logits_t^{(L)}}$
  - 回忆一下普通神经网络时代的定义：

$$\delta^{(L)} = \nabla_y Cost \odot \sigma'(logit^{(L)}) \quad (BP1)$$
$$\delta^{(l)} = ((w^{(l+1)})^T \delta^{(l+1)}) \odot \sigma'(logit^{(l)}) \quad (BP2)$$

，且

$$o_t = g(Vs_t + b_o) \quad (1)$$
$$s_t = f(Ux_t + Ws_{t-1} + b_s) \quad (2)$$



$\delta_t^{(L)}$ 可以由各个时刻的o_t与ground truth计算得出

$\delta_t^{(L-1)}$ 可以由（1）计算得出，仅与V有关，和普通神经网络一样

我们将（2）分解为： $logit_t = Ux_t + Ws_{t-1}$，$s_{t-1} = f(logit_{t-1})$ 或写成 $z_t = Ux_t + Ws_{t-1}$，$s_{t-1} = f(z_{t-1})$

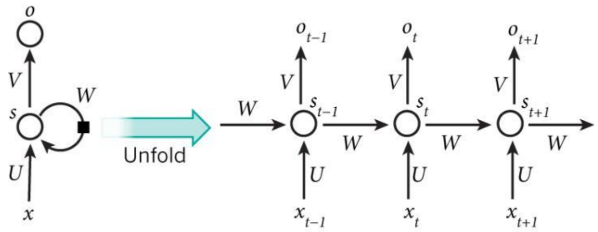
$\delta_{t-1}^{(L-1)}$ 可以由 $\delta_t^{(L-1)}$ 沿时间方向向前传播得出

▶循环神经网络的反向传播

$$\delta^{(L)}_{(T)} = \nabla_y Cost \odot \sigma'(logit^{(L)}_{(T)}) \qquad (BP1)$$

$$\delta^{(l)}_{(T)} = ((V^{(l+1)})^T \delta^{(l+1)}_{(T)}) \odot \sigma'(logit^{(l)}_{(T)}) \qquad (BP2)$$

$$\delta^{(l)}_{(t)} = ((W^{(l)})^T \delta^{(l)}_{(t+1)} + (V^{(l+1)})^T \delta^{(l+1)}_{(t)}) \odot \sigma'(s^{(l)}_{(t)}) \qquad (BP2T)$$



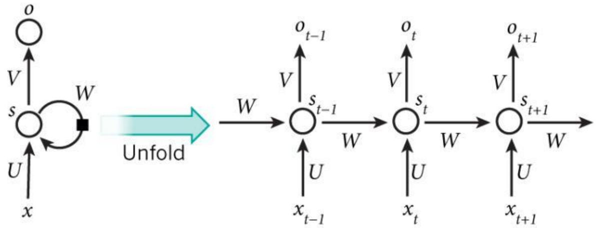
$$\frac{\partial Cost}{\partial bias^{(l)}} = \sum_{t=0}^{T} \delta^{(l)}_{(t)} \qquad (BP3)$$

$$\frac{\partial Cost}{\partial V^{(l)}} = \sum_{t=0}^{T} \delta^{(l)}_{(t)} \cdot (s^{(l-1)}_{(t)})^T \qquad (BP4V)$$

$$\frac{\partial Cost}{\partial W^{(l)}} = \sum_{t=1}^{T} \delta^{(l)}_{(t)} \cdot (s^{(l)}_{(t-1)})^T \qquad (BP4W)$$

$$\frac{\partial Cost}{\partial U^{(l)}} = \sum_{t=0}^{T} \delta^{(l)}_{(t)} \cdot (x_{(t)})^T \qquad (BP4U)$$

▶ 循环神经网络的反向传播



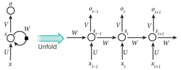
$$o_t = g(Vs_t + b_o) \tag{1}$$
$$s_t = f(Ux_t + Ws_{t-1} + b_s) \tag{2}$$

$$U = \begin{bmatrix} 0.10 & 0.20 & 0.30 \\ 0.40 & 0.50 & 0.60 \end{bmatrix}, \quad W = \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix}, V = \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix}, b_s = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} b_o = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix}$$

$$x_1 = \begin{bmatrix} 1.00 \\ 2.00 \\ 3.00 \end{bmatrix}, x_2 = \begin{bmatrix} 2.00 \\ 3.00 \\ 4.00 \end{bmatrix} x_3 = \begin{bmatrix} 3.00 \\ 4.00 \\ 5.00 \end{bmatrix}$$

$$GroundTruth_1 = \begin{bmatrix} 1.00 \\ 0.00 \\ 0.00 \end{bmatrix} GroundTruth_2 = \begin{bmatrix} 0.00 \\ 0.00 \\ 1.00 \end{bmatrix} GroundTruth_3 = \begin{bmatrix} 0.00 \\ 1.00 \\ 0.00 \end{bmatrix}$$

循环神经网络的反向传播



$$o_t = g(Vs_t + b_o) \tag{1}$$
$$s_t = f(Ux_t + Ws_{t-1} + b_s) \tag{2}$$

$$U = \begin{bmatrix} 0.10 & 0.20 & 0.30 \\ 0.40 & 0.50 & 0.60 \end{bmatrix}, \quad W = \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix}, V = \begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix}, b_s = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix} b_o = \begin{bmatrix} 0.00 \\ 0.00 \end{bmatrix}$$

$$x_1 = \begin{bmatrix} 1.00 \\ 2.00 \\ 3.00 \end{bmatrix}, x_2 = \begin{bmatrix} 2.00 \\ 3.00 \\ 4.00 \end{bmatrix} x_3 = \begin{bmatrix} 3.00 \\ 4.00 \\ 5.00 \end{bmatrix}$$

$$GroundTruth_1 = \begin{bmatrix} 1.00 \\ 0.00 \\ 0.00 \end{bmatrix} GroundTruth_2 = \begin{bmatrix} 0.00 \\ 0.00 \\ 1.00 \end{bmatrix} GroundTruth_3 = \begin{bmatrix} 0.00 \\ 1.00 \\ 0.00 \end{bmatrix}$$

$$\delta^{(L)}_{(T)} = \nabla_y Cost \odot \sigma'(logit^{(L)}_{(T)}) \tag{BP1}$$
$$\delta^{(l)}_{(T)} = ((V^{(l+1)})^T \delta^{(l+1)}_{(T)}) \odot \sigma'(logit^{(l)}_{(T)}) \tag{BP2}$$
$$\delta^{(l)}_{(t)} = ((W^{(l)})^T \delta^{(l)}_{(t+1)} + (V^{(l+1)})^T \delta^{(l+1)}_{(t)}) \odot \sigma'(s^{(l)}_{(t)}) \tag{BP2T}$$
$$\frac{\partial Cost}{\partial bias^{(l)}} = \sum_{t=0}^{T} \delta^{(l)}_{(t)} \tag{BP3}$$
$$\frac{\partial Cost}{\partial V^{(l)}} = \sum_{t=0}^{T} \delta^{(l)}_{(t)} \cdot (s^{(l-1)}_{(t)})^T \tag{BP4V}$$
$$\frac{\partial Cost}{\partial W^{(l)}} = \sum_{t=1}^{T} \delta^{(l)}_{(t)} \cdot (s^{(l)}_{(t-1)})^T \tag{BP4W}$$
$$\frac{\partial Cost}{\partial U^{(l)}} = \sum_{t=0}^{T} \delta^{(l)}_{(t)} \cdot (x_{(t)})^T \tag{BP4U}$$

$$\delta^3_3 = \nabla_y Cost \odot \sigma'(logit^{(L)}_{(T)}) = \begin{bmatrix} 0.21231 \\ -0.68366 \\ 0.47135 \end{bmatrix}$$

$$\nabla V_3 = \delta^3_3 \cdot (s_3)^T = \begin{bmatrix} 0.21231 \\ -0.68366 \\ 0.47135 \end{bmatrix} \cdot \begin{bmatrix} 0.99394 & 1.00000 \end{bmatrix} = \begin{bmatrix} 0.21 & 0.21 \\ -0.68 & -0.68 \\ 0.47 & 0.47 \end{bmatrix}$$

$$\delta^3_2 = \nabla_y Cost \odot \sigma'(logit^{(L)}_{(T)}) = \begin{bmatrix} 0.21308 \\ 0.31658 \\ -0.52965 \end{bmatrix}$$

$$\nabla V_2 = \delta^3_2 \cdot (s_2)^T = \begin{bmatrix} 0.21308 \\ 0.31658 \\ -0.52965 \end{bmatrix} \cdot \begin{bmatrix} 0.97961 & 0.99996 \end{bmatrix} = \begin{bmatrix} 0.21 & 0.21 \\ 0.31 & 0.32 \\ -0.52 & -0.53 \end{bmatrix}$$

$$\delta^3_1 = \nabla_y Cost \odot \sigma'(logit^{(L)}_{(T)}) = \begin{bmatrix} -0.78166 \\ 0.31813 \\ 0.46353 \end{bmatrix}$$

$$\nabla V_1 = \delta^3_1 \cdot (s_1)^T = \begin{bmatrix} -0.78166 \\ 0.31813 \\ 0.46353 \end{bmatrix} \cdot \begin{bmatrix} 0.88535 & 0.99668 \end{bmatrix} = \begin{bmatrix} -0.69 & -0.78 \\ 0.28 & 0.32 \\ 0.41 & 0.46 \end{bmatrix}$$

$$\delta^2_3 = (W^T \cdot \delta^2_4 + V^T \cdot \delta^3_3) \odot \sigma'(s^2_3)$$
$$= ((\begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix})^T \cdot \begin{bmatrix} 0.00000 \\ 0.00000 \end{bmatrix} + (\begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix})^T \cdot \begin{bmatrix} 0.21231 \\ -0.68366 \\ 0.47135 \end{bmatrix}) \odot \begin{bmatrix} 0.01209 \\ 0.00000 \end{bmatrix}$$
$$= \begin{bmatrix} 0.00063 \\ 0.00000 \end{bmatrix}$$

$$\delta^2_2 = (W^T \cdot \delta^2_3 + V^T \cdot \delta^3_2) \odot \sigma'(s^2_2)$$
$$= ((\begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix})^T \cdot \begin{bmatrix} 0.00063 \\ 0.00000 \end{bmatrix} + (\begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix})^T \cdot \begin{bmatrix} 0.21308 \\ 0.31658 \\ -0.52965 \end{bmatrix}) \odot \begin{bmatrix} 0.04036 \\ 0.00009 \end{bmatrix}$$
$$= \begin{bmatrix} -0.00599 \\ -0.00001 \end{bmatrix}$$

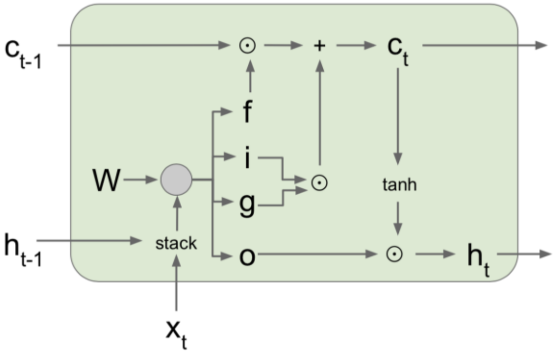
$$\delta^2_1 = (W^T \cdot \delta^2_2 + V^T \cdot \delta^3_1) \odot \sigma'(s^2_1)$$
$$= ((\begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \end{bmatrix})^T \cdot \begin{bmatrix} -0.00599 \\ -0.00001 \end{bmatrix} + (\begin{bmatrix} 0.10 & 0.20 \\ 0.30 & 0.40 \\ 0.50 & 0.60 \end{bmatrix})^T \cdot \begin{bmatrix} -0.78166 \\ 0.31813 \\ 0.46353 \end{bmatrix}) \odot \begin{bmatrix} 0.21615 \\ 0.00662 \end{bmatrix}$$
$$= \begin{bmatrix} 0.05370 \\ 0.00164 \end{bmatrix}$$

$$\nabla W = \sum_{t=1}^{3} \delta^{(2)}_{(t)} \cdot (s^{(2)}_{(t-1)})^T$$
$$= \begin{bmatrix} 0.00061 & 0.00063 \\ 0.00000 & 0.00000 \end{bmatrix} + \begin{bmatrix} -0.00531 & -0.00597 \\ -0.00001 & -0.00001 \end{bmatrix}$$
$$= \begin{bmatrix} -0.00469 & -0.00535 \\ -0.00001 & -0.00001 \end{bmatrix}$$

$$\nabla V = \sum_{t=0}^{3} \delta^{(3)}_{(t)} \cdot (s^{(2)}_{(t)})^T$$
$$= \begin{bmatrix} -0.27229 & -0.35369 \\ -0.08774 & -0.05002 \\ 0.36003 & 0.40372 \end{bmatrix}$$

$$\nabla U = \sum_{t=0}^{3} \delta^{(1)}_{(t)} \cdot (x_{(t)})^T$$
$$= \begin{bmatrix} -0.27229 & -0.35369 \\ -0.08774 & -0.05002 \\ 0.36003 & 0.40372 \end{bmatrix}$$

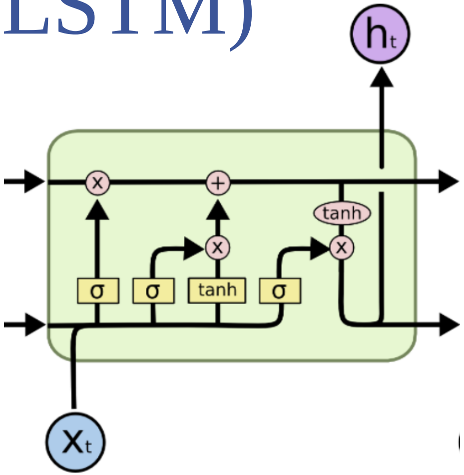
Long Short-Term Memory(LSTM)

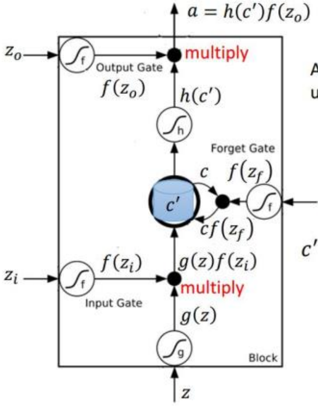


$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$



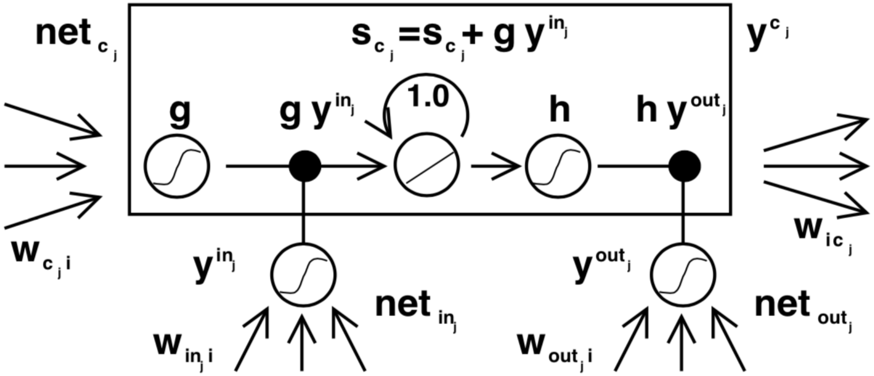


Activation function f is usually a sigmoid function

Between 0 and 1

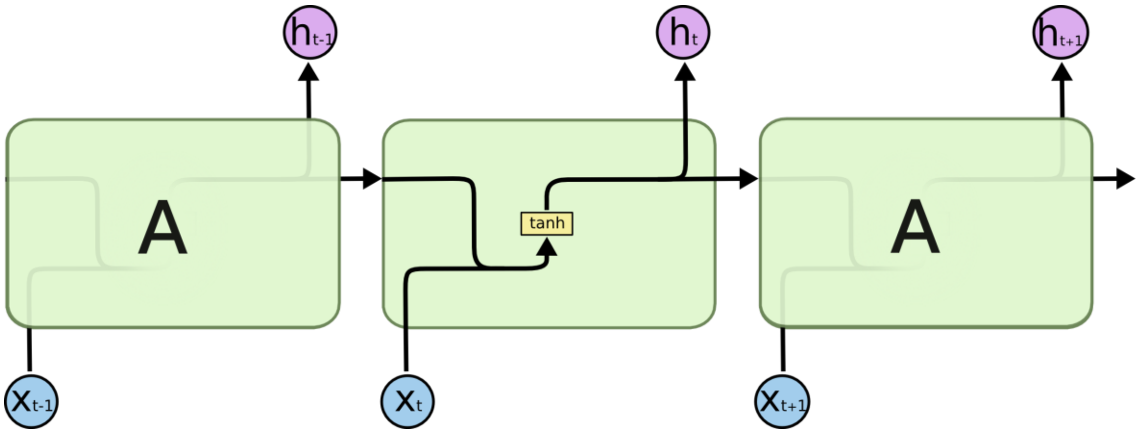
Mimic open and close gate

$$c' = g(z)f(z_i) + cf(z_f)$$

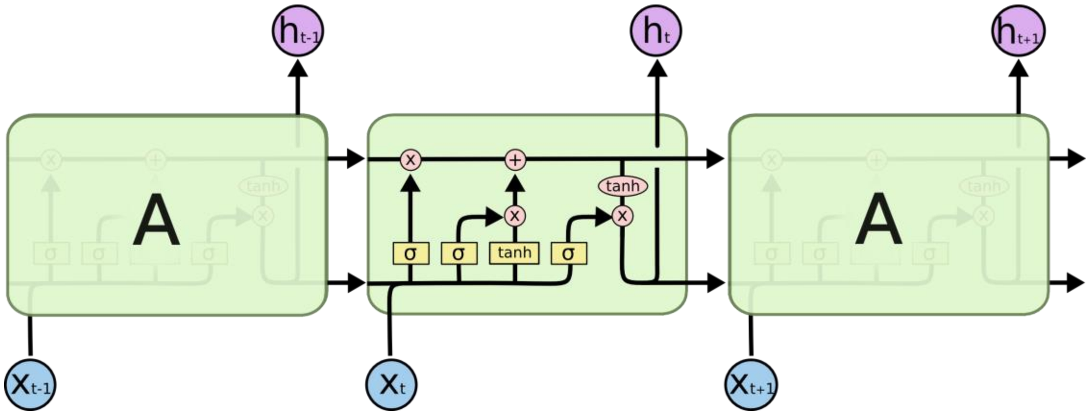


Long Short-Term Memory(LSTM)

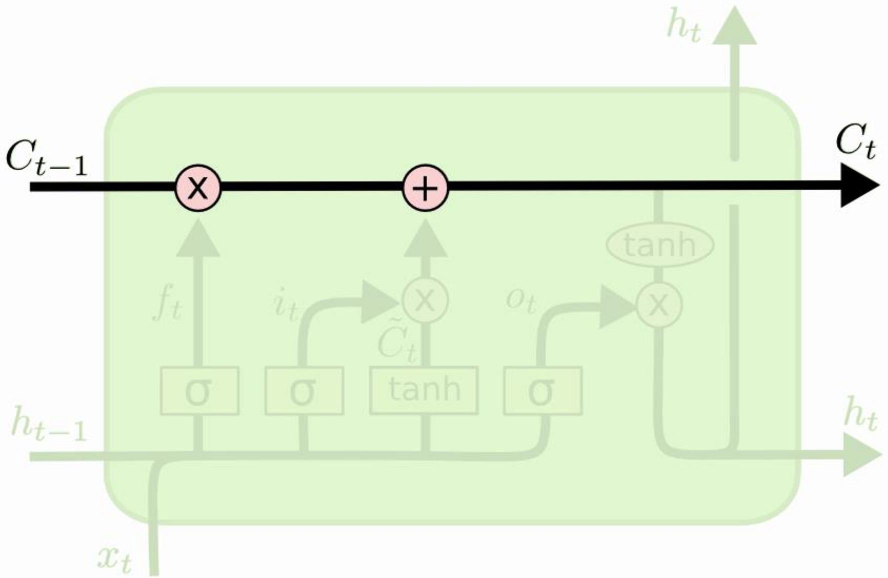
RNN:



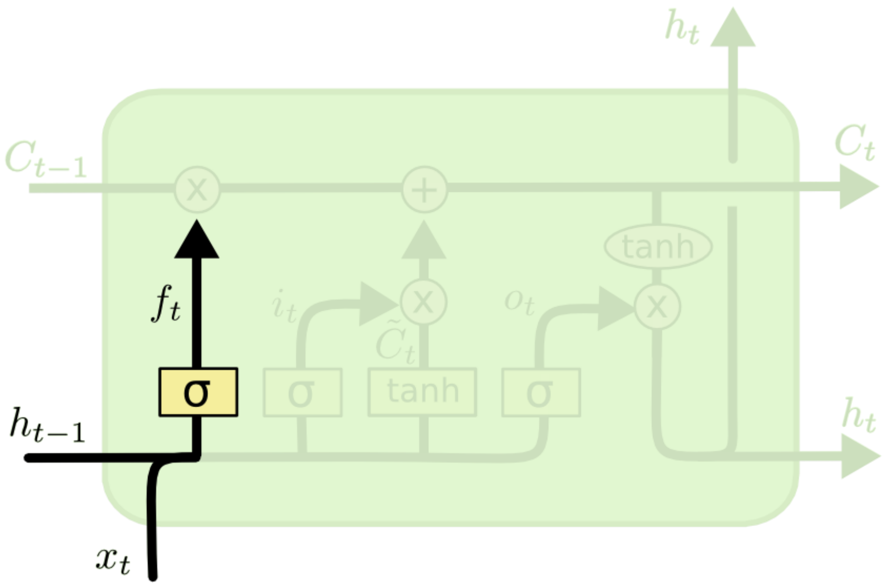
LSTM:



▶ Long Short-Term Memory(LSTM)

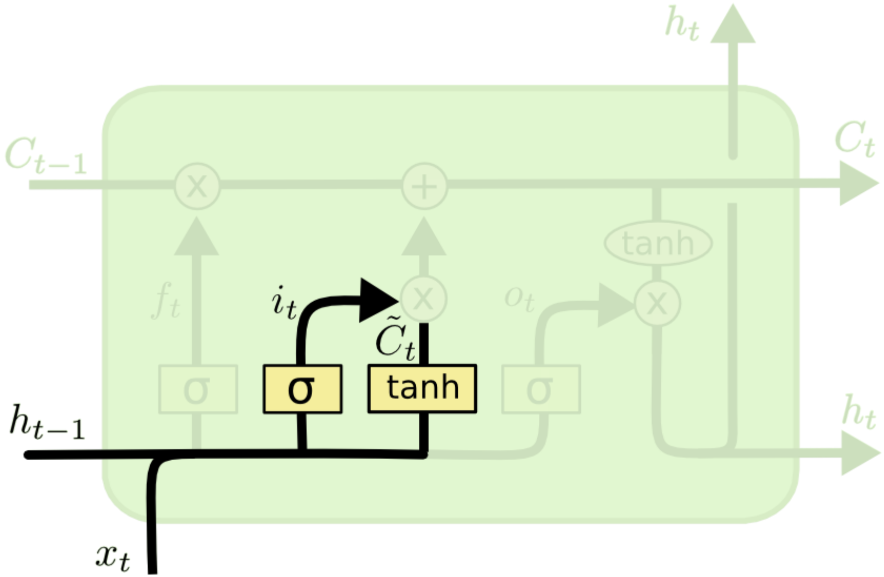


Long Short-Term Memory(LSTM)



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

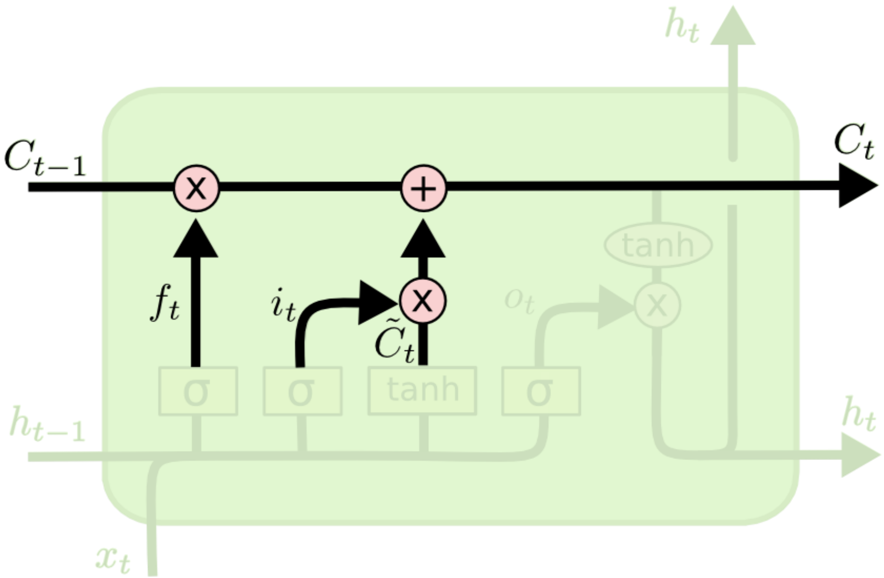
Long Short-Term Memory(LSTM)



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

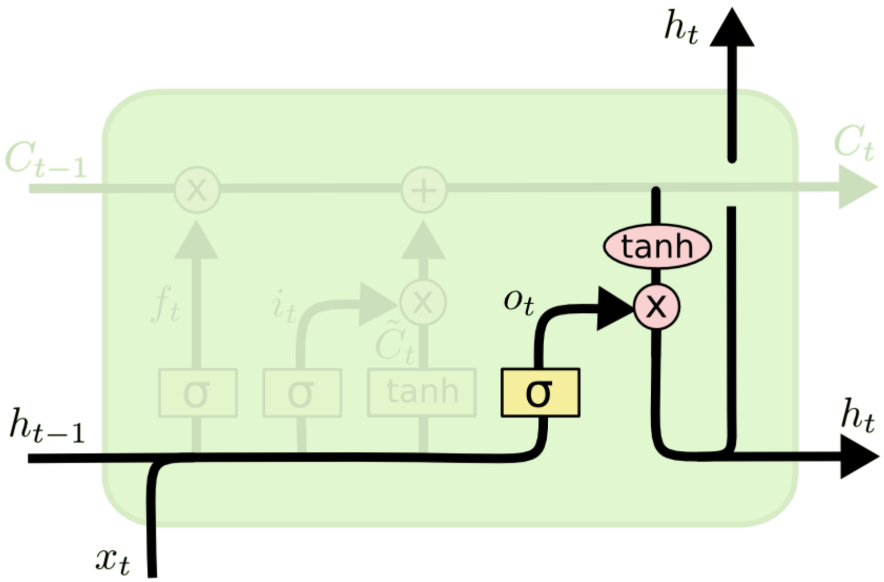
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Long Short-Term Memory(LSTM)



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

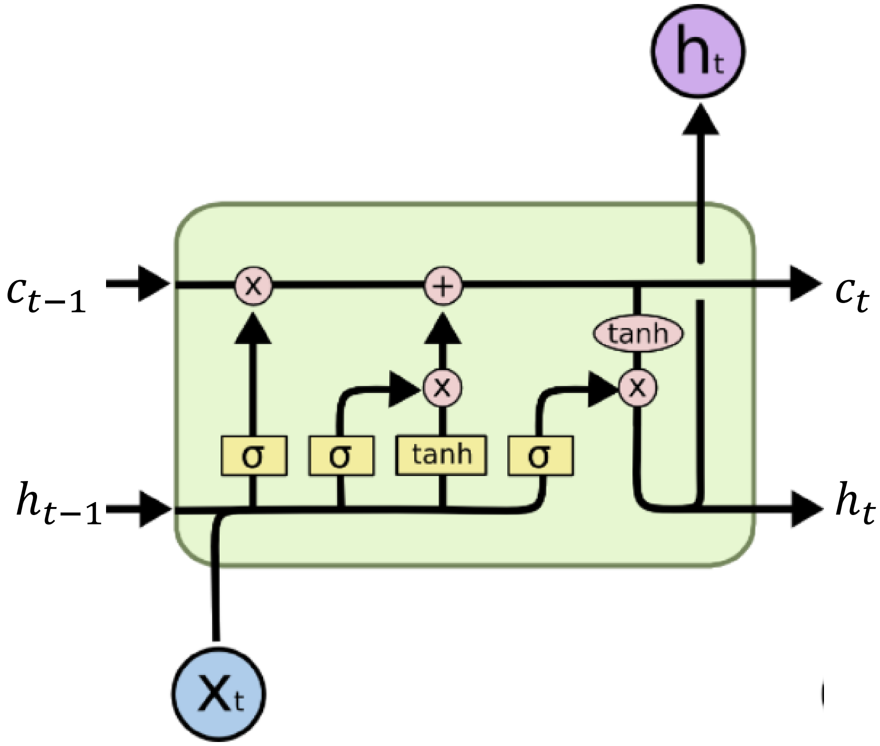
Long Short-Term Memory(LSTM)



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

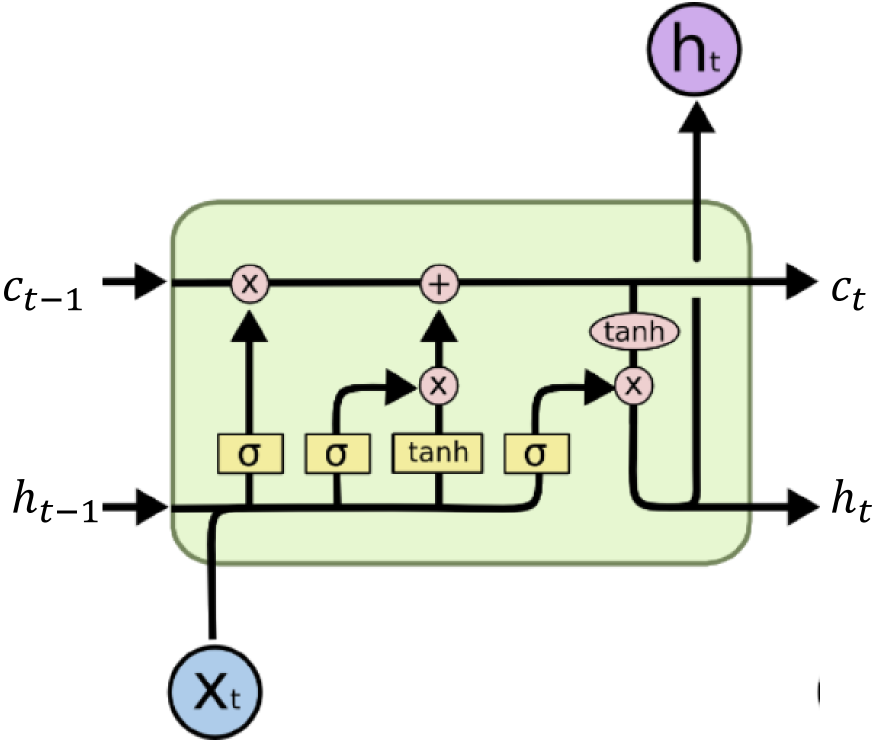
$$h_t = o_t * \tanh\left(C_t\right)$$

Long Short-Term Memory(LSTM)



$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{1}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2}$$
$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{3}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{4}$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5}$$
$$h_t = o_t \odot \tanh(c_t) \tag{6}$$

Long Short-Term Memory(LSTM)



$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{1}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2}$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{3}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{4}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t \odot \tanh(c_t) \tag{6}$$

$$\delta_t^{(L)} = \frac{\partial Cost}{\partial logits_t^{(L)}}$$

Word Embedding

- 词嵌入表示
  - Efficient Estimation of Word Representation in Vector Space（2013）
- 什么是词嵌入？

一直以来，我们都将数据以one-hot向量的形式输入

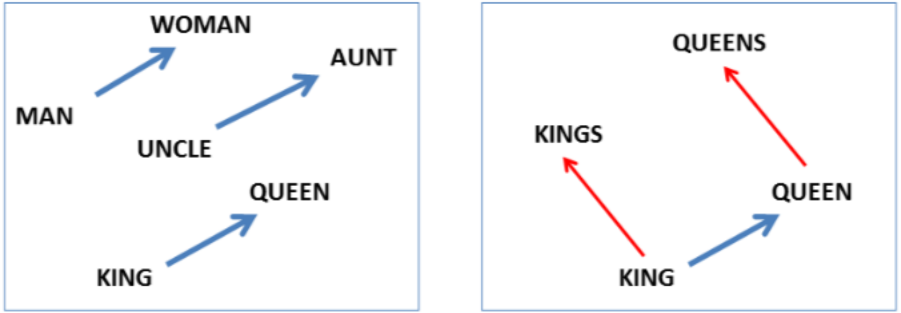
$$我=\begin{bmatrix}1\\0\\0\\0\end{bmatrix}，爱=\begin{bmatrix}0\\1\\0\\0\end{bmatrix}，学=\begin{bmatrix}0\\0\\1\\0\end{bmatrix}，习=\begin{bmatrix}0\\0\\0\\1\end{bmatrix} \qquad 深度=\begin{bmatrix}0\\\cdots\\1\\\cdots\\0\end{bmatrix}，学习=\begin{bmatrix}0\\\cdots\\1\\\cdots\\0\end{bmatrix}$$

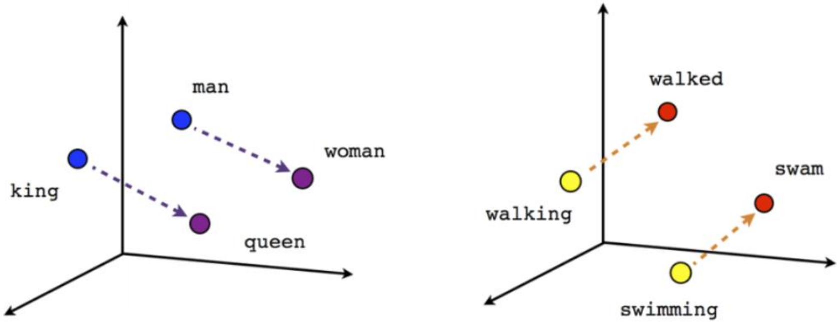
- 计算量大
- 词之间的关联无法体现

$$深度=\begin{bmatrix}0.92\\0.14\\0.37\\0.88\\-0.03\end{bmatrix}，学习=\begin{bmatrix}0.11\\0.52\\-0.82\\0.004\\0.02\end{bmatrix}$$

Word Embedding

- 词向量在语义上的优势





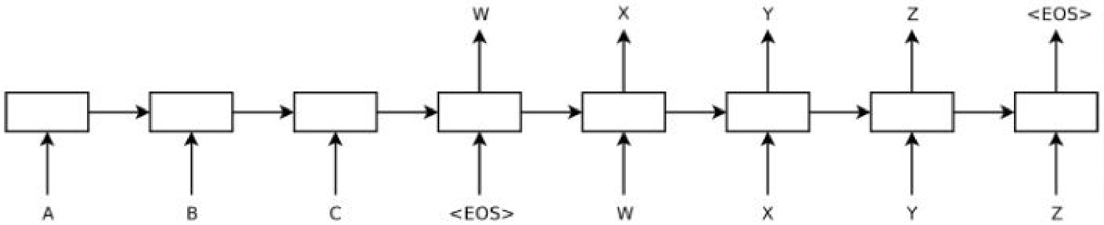
Word Embedding

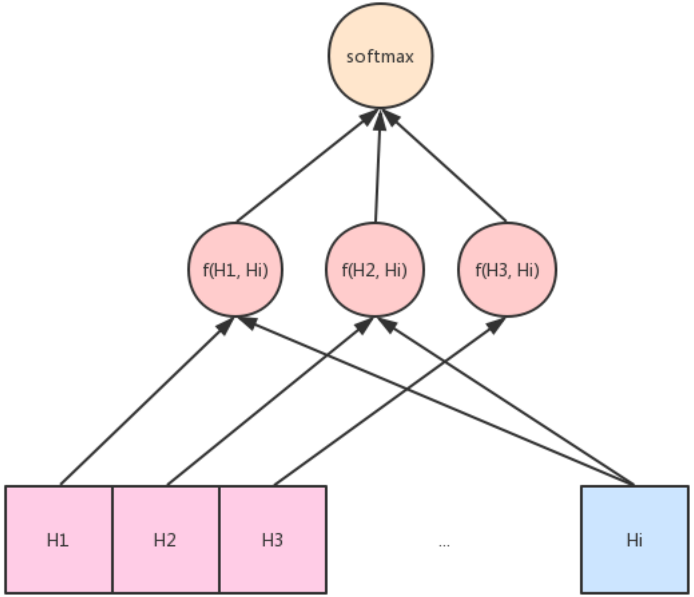
- Word2Vec训练
  - Skip-gram
  - CBOW
- 使用最简单的神经网络
- 看起来好像是在预测和输入词关联的词
- 但是我们需要的其实只是权重矩阵本身
  - 例如：输入数据： “我爱学习爱劳动，是个好青年”
    →我爱学习爱劳动是个好青年

    →(我,爱），（我,学），（爱,我），（爱,学），（爱,习），（学,我），…

$$我=\begin{bmatrix}1\\0\\\cdots\\0\end{bmatrix}^T\cdot\begin{bmatrix}0.75 & 0.82\\0.43 & -0.81\\\cdots & \cdots\\-0.2 & 0.9\end{bmatrix}=[0.75\ \ 0.82]$$

Attention Model





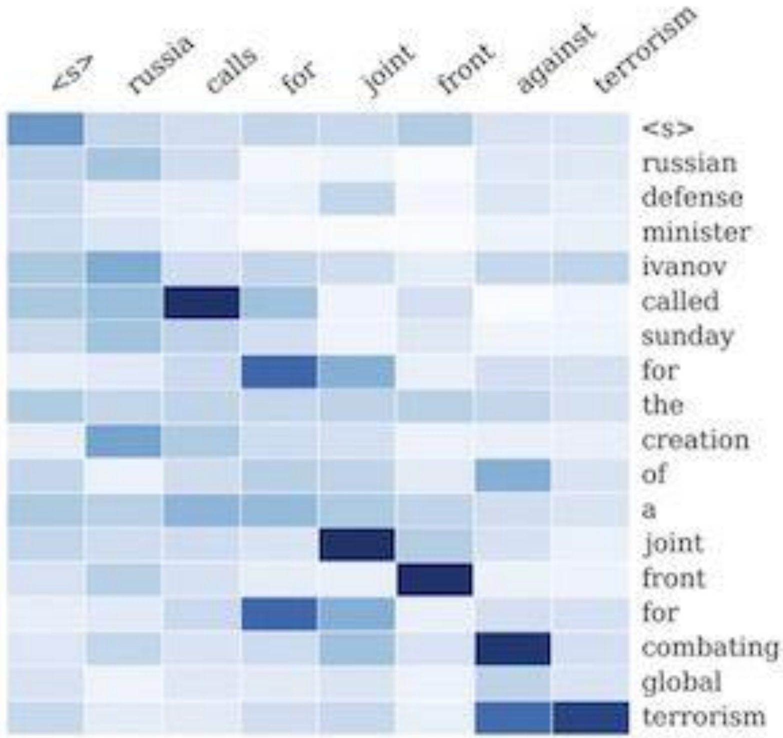


Figure 1: Example output of the attention-based summarization (ABS) system. The heatmap represents a soft alignment between the input (right) and the generated summary (top). The columns represent the distribution over the input after generating each word.

THANK YOU