第九讲 循环神经网络 II

北京交通大学 《深度学习》课程组

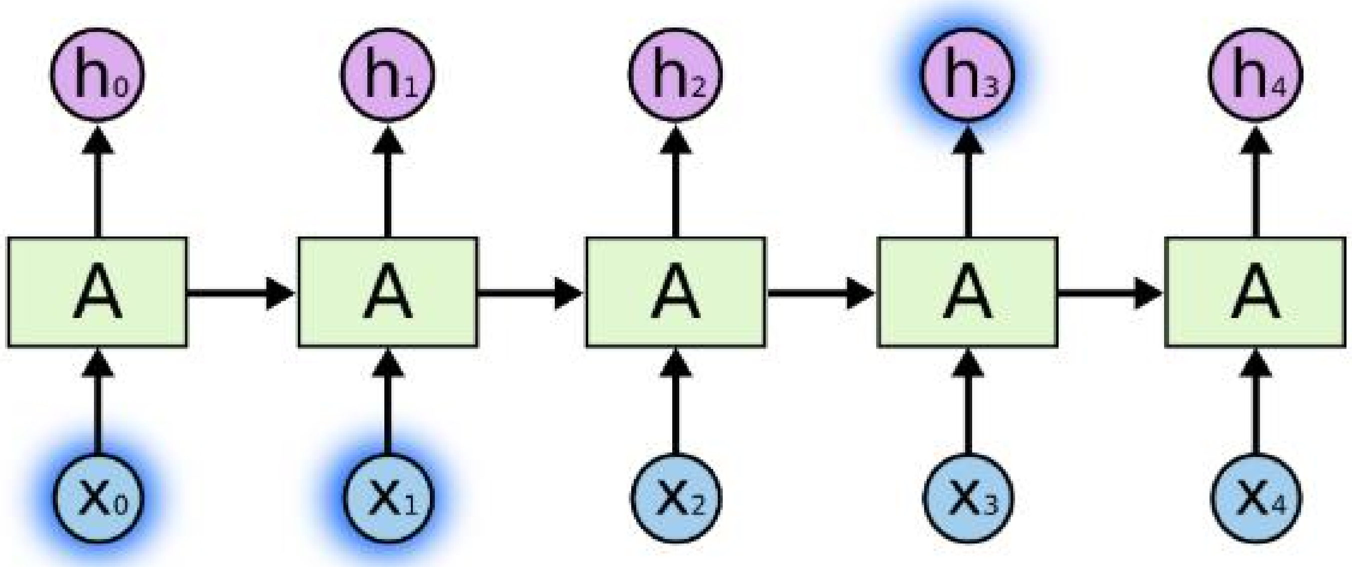
目录

目录

9.1 长程依赖问题

RNN 的长处之一是它可以在执行当前任务时利用先前的信息，尤其当相关信息和预测信息间隔较小时，效果更佳明显。

如预测句子“the clouds are in the *sky*”中的最后一个词。



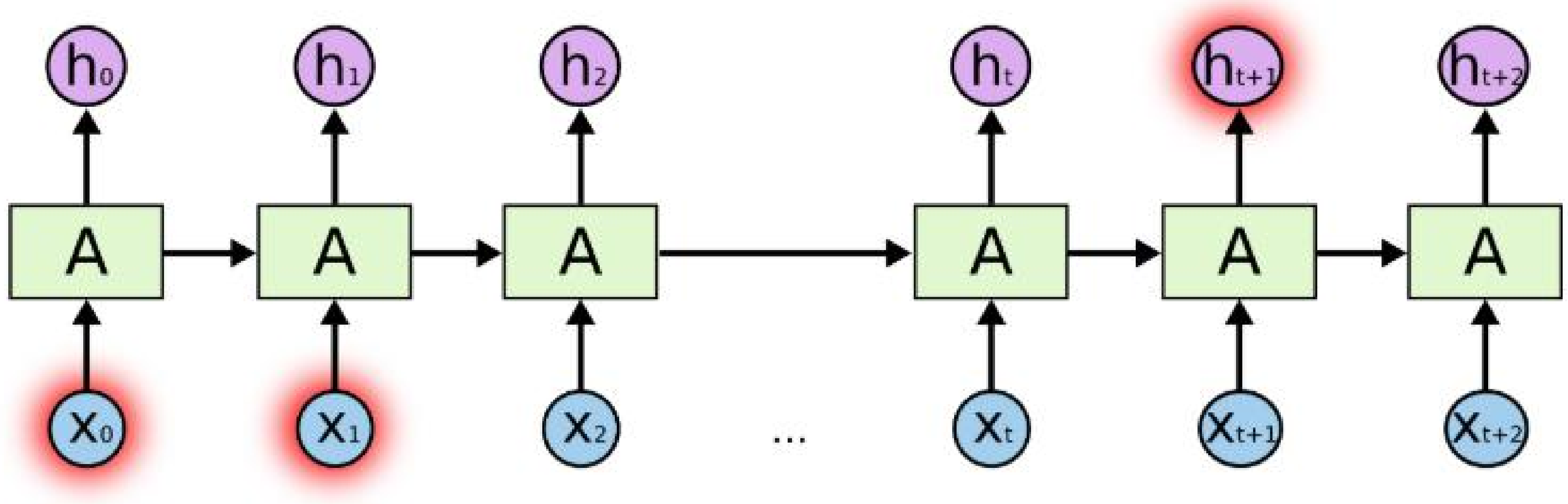
较近的相关信息或位置间隔

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

9.1 长程依赖问题

但在间隔不断增大时，RNN 会丧失学习到连接如此远的信息的能力。

如预测句子“I grew up in France… I speak fluent *French*”中最后一个词。



较长的相关信息或位置间隔

Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult[J]. IEEE transactions on neural networks, 1994, 5(2): 157-166.

9.1 长程依赖问题

为什么在实际应用中，RNN很难处理长距离的依赖？

上一节RNN推导中，误差项沿时间反向传播的方法：

$$\delta_k^{\mathrm{T}} = \delta_t^{\mathrm{T}} \prod_{i=k}^{t-1} \operatorname{diag}\left[f'\left(\mathbf{net}_i\right)\right] W$$

$$\begin{aligned}
\delta_k^{\mathrm{T}} &= \frac{\partial E}{\partial \mathrm{net}_k} \\
&= \frac{\partial E}{\partial \mathrm{net}_t} \frac{\partial \mathrm{net}_t}{\partial \mathrm{net}_k} \\
&= \frac{\partial E}{\partial \mathrm{net}_t} \frac{\partial \mathrm{net}_t}{\partial \mathrm{net}_{t-1}} \frac{\partial \mathrm{net}_{t-1}}{\partial \mathrm{net}_{t-2}} \cdots \frac{\partial \mathrm{net}_{k+1}}{\partial \mathrm{net}_k} \\
&= W \operatorname{diag}[f'(\mathrm{net}_{t-1})] W \operatorname{diag}[f'(\mathrm{net}_{t-2})] \ldots W \operatorname{diag}[f'(\mathrm{net}_k)] \delta_t^{\mathrm{T}} \\
&= \delta_t^{\mathrm{T}} \prod_{i=k}^{t-1} W \operatorname{diag}[f'(\mathrm{net}_i)]
\end{aligned}$$

根据下面的不等式，来获取 δ_k^{T} 模的上界（模可以看做对δ_k^{T} 中每一项值大小的度量）：

$$\begin{aligned}
\left\|\delta_k^{\mathrm{T}}\right\| &\leqslant \left\|\delta_t^{\mathrm{T}}\right\| \prod_{i=k}^{t-1} \left\|\operatorname{diag}[f'(\mathbf{net}_i)]\right\| \|W\| \\
&\leqslant \left\|\delta_t^{\mathrm{T}}\right\| (\beta_f \beta_W)^{t-k}
\end{aligned}$$

https://zybuluo.com/hanbingtao/note/541458

9.1 长程依赖问题

$$\left\|\delta_k^{\mathrm{T}}\right\| \leqslant \left\|\delta_t^{\mathrm{T}}\right\| \prod_{i=k}^{t-1} \left\|\operatorname{diag}\left[f'(\mathbf{net}_i)\right]\right\| \|W\| \leqslant \left\|\delta_t^{\mathrm{T}}\right\| \left(\beta_f \beta_W\right)^{t-k}$$

可以看到，误差项从t时刻传递到k时刻，其值的上界是 $\beta_f\beta_w$ 的指数函数。$\beta_f\beta_w$ 分别是对角矩阵和矩阵W模的上界。

当t-k很大时(也就是误差传递很多个时刻时), 整个式子的值就会变得极小（当 $\beta_f\beta_w$ 乘积小于1）或者极大(当 $\beta_f\beta_w$ 乘积大于1)，前者是梯度消失，后者是梯度爆炸。

梯度消失或者梯度爆炸会导致梯度为0或NaN，进而无法继续训练更新参数。上述现象称之为RNN长程依赖问题。

https://zybuluo.com/hanbingtao/note/541458

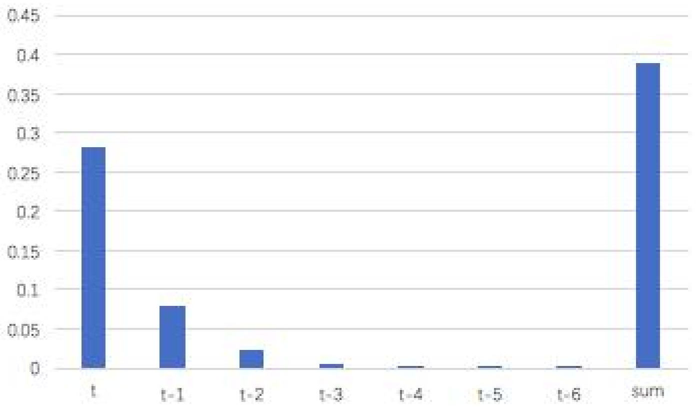
梯度消失举例

RNN中权重数组W最终的梯度是各个时刻的梯度之和，即：

$$\nabla_W E = \sum_{k=1}^{t} \nabla_{Wk} E$$
$$= \nabla_{Wt} E + \nabla_{Wt-1} E + \nabla_{Wt-2} E + \ldots + \nabla_{W1} E$$

$$\begin{aligned}\nabla_W E &= \frac{\partial E}{\partial W} \\ &= \frac{\partial E}{\partial \mathrm{net}_t}\frac{\partial \mathrm{net}_t}{\partial W} \\ &= \nabla_{Wt} E + \delta_{t-1}^T \frac{\partial \mathrm{net}_{t-1}}{\partial W} \\ &= \nabla_{Wt} E + \nabla_{Wt-1} E + \delta_{t-2}^T \frac{\partial \mathrm{net}_{t-2}}{\partial W} \\ &= \nabla_{Wt} E + \nabla_{Wt-1} E + \ldots + \nabla_{W1} E \\ &= \sum_{k=1}^{t} \nabla_{Wk} E\end{aligned}$$

假设某轮训练中，各时刻的梯度以及最终的梯度之和如下图：



从t-3时刻开始，梯度已经几乎减少到0了。即从此时刻开始再往之前走，得到的梯度（几乎为零）就不会对最终的梯度值有任何贡献。这就是**原始RNN无法处理长距离依赖的原因。**

9.1 长程依赖问题

通常来说，**梯度爆炸**更容易处理一些。因为梯度爆炸的时候，程序会收到NaN错误。也可以设置一个梯度阈值，当梯度超过这个阈值时直接截取。

梯度消失更难检测，更难处理。通常有三种缓解梯度消失的方法：

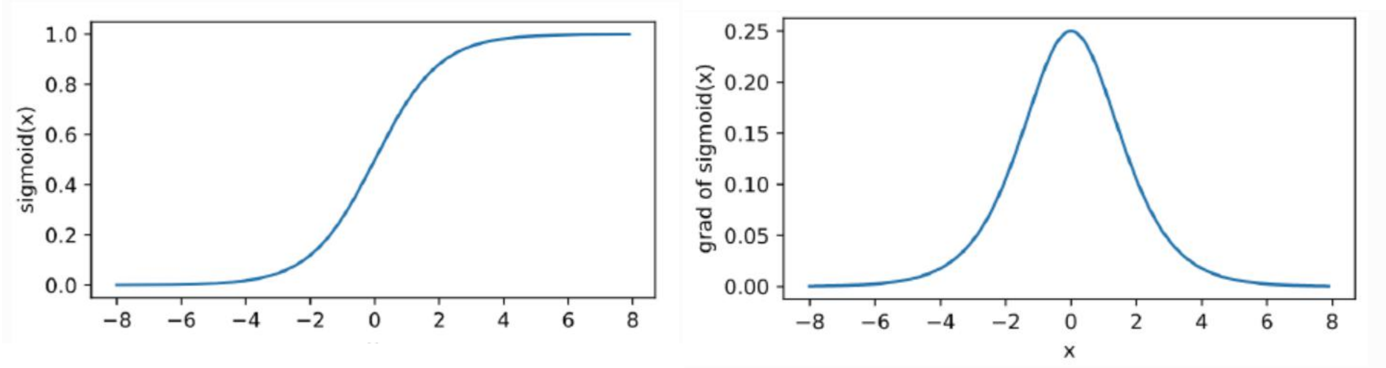
1. 初始化合理的权重值：初始化权重，使每个神经元尽可能躲开梯度消失的区域，如避免取极大或极小值。

2. 选取合适的激活函数：如用relu代替sigmoid和tanh。

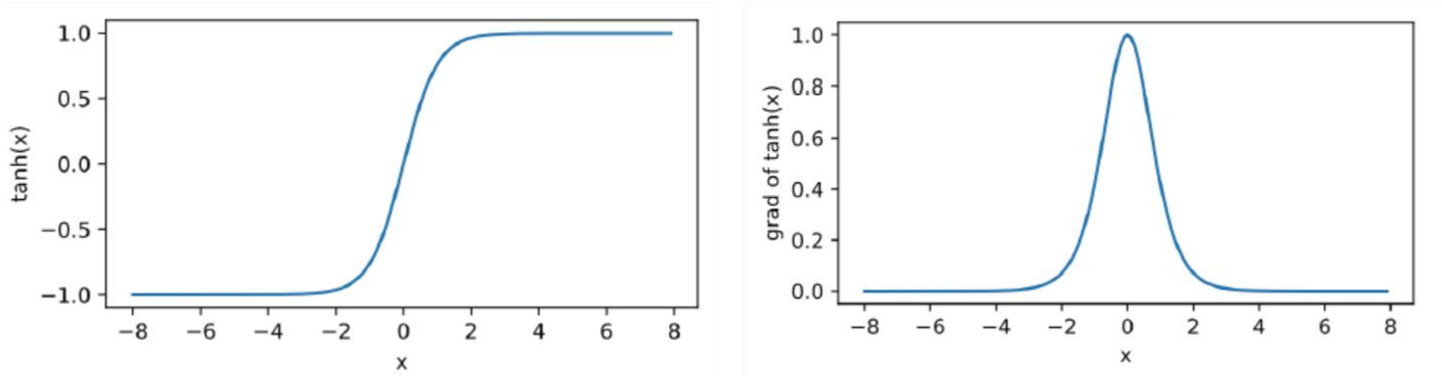
3. 引入合理的记忆网络：如长短时记忆网络（LTSM）、Gated Recurrent Unit（GRU）等。

9.1 长程依赖问题

sigmoid函数的函数图和导数图



tanh函数的函数图和导数图

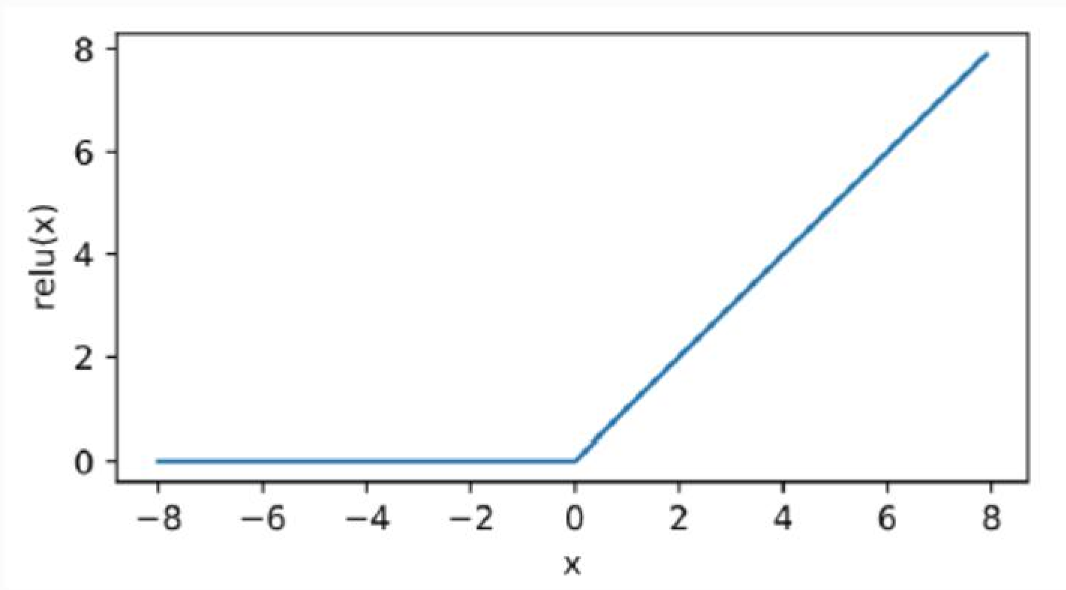


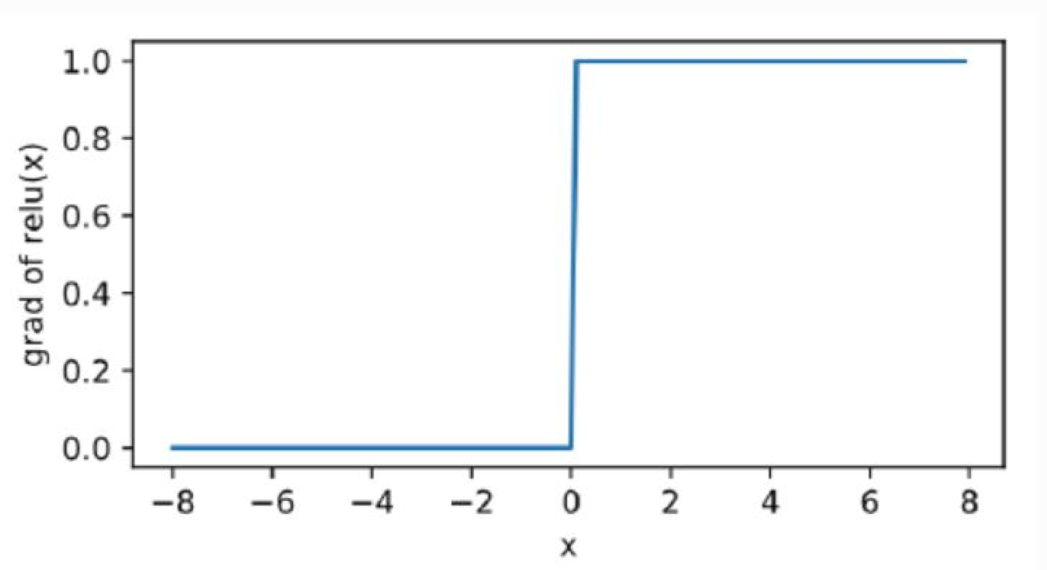
sigmoid函数与tanh函数比较：

- sigmoid函数的导数值范围为(0,0.25]，反向传播时会导致梯度消失
- tanh函数的导数值范围为(0,1]，相对范围较大，但仍会导致梯度消失
- sigmoid函数不是0中心对称，输出均大于0
- tanh函数是0中心对称，可以使网络收敛的更好

9.1 长程依赖问题

ReLU函数的图像和导数图为

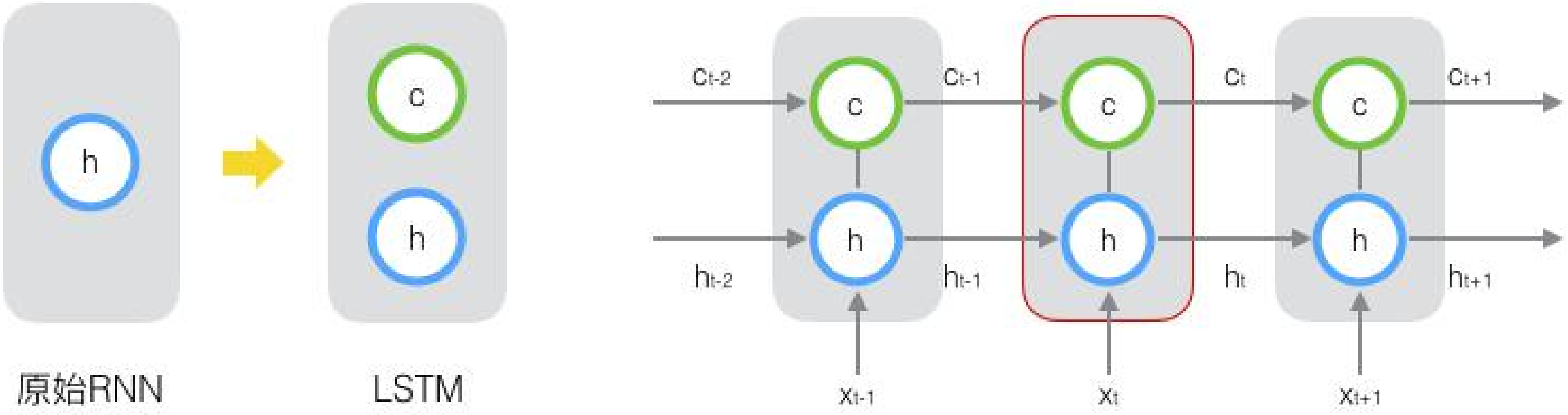




ReLU函数的左侧导数为0，右侧导数恒为1，避免了小数的连乘，但反向传播中仍有权值的累乘。ReLU函数改善了“梯度消失”现象。

目录

Long Short-Term Memory networks（以下简称LSTMs），一种特殊的RNN网络，该网络正是为了解决长程依赖问题。



增加状态c，称为单元状态(cell state)，用以保存长期状态

Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.

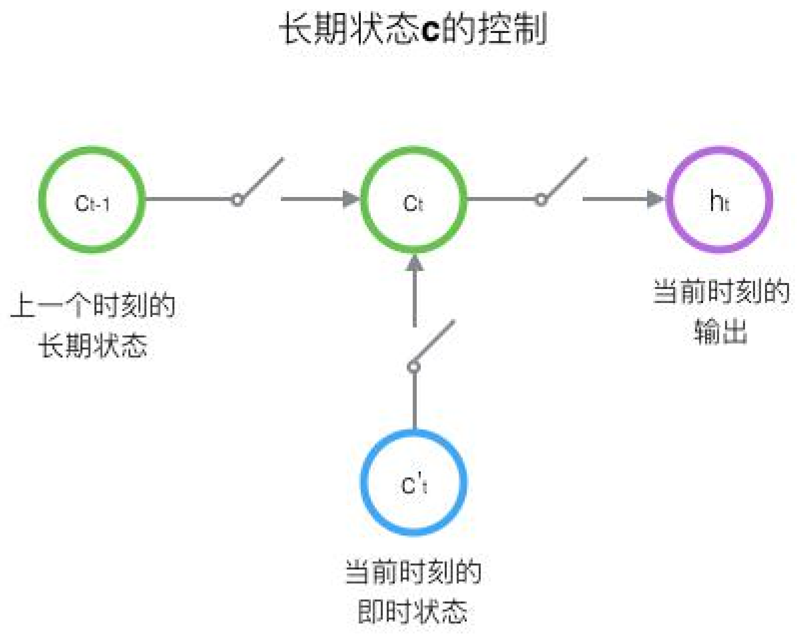
9.2 长短期记忆网络（LSTM）

LSTM的关键是如何控制长期状态c，为此引入三个控制开关：

第一个开关，负责控制如何继续保存长期状态c；

第二个开关，负责控制把即时状态输入到长期状态c；

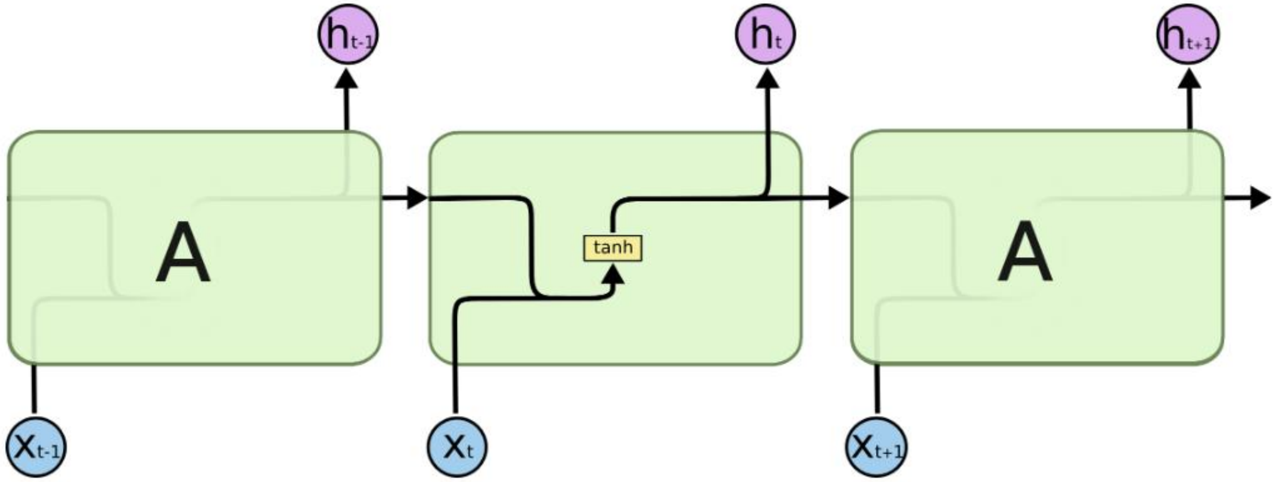
第三个开关，负责控制是否把长期状态c作为当前的LSTM的输出



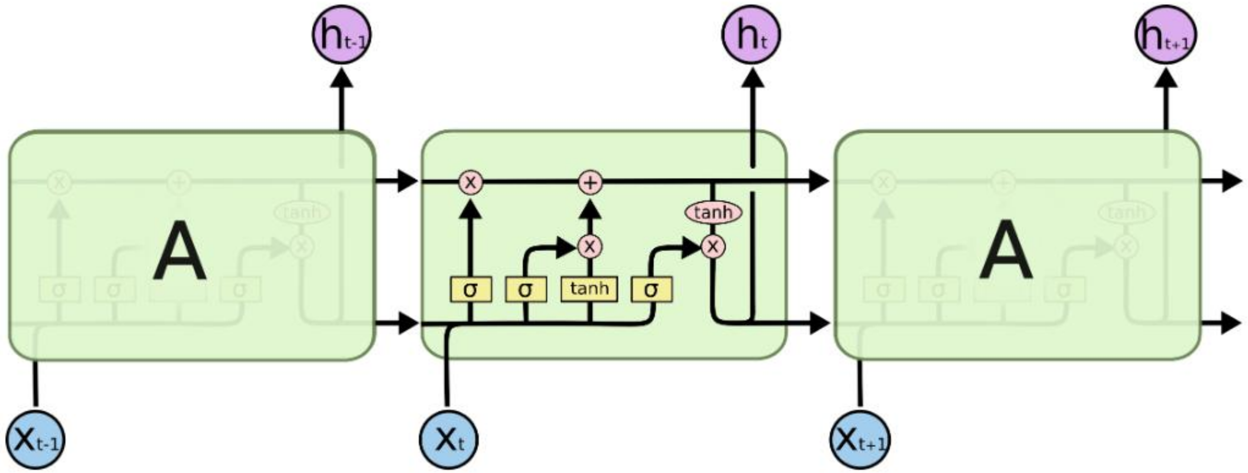
https://zhuanlan.zhihu.com/p/32481747

9.2 长短期记忆网络（LSTM）

标准RNN的重复模块



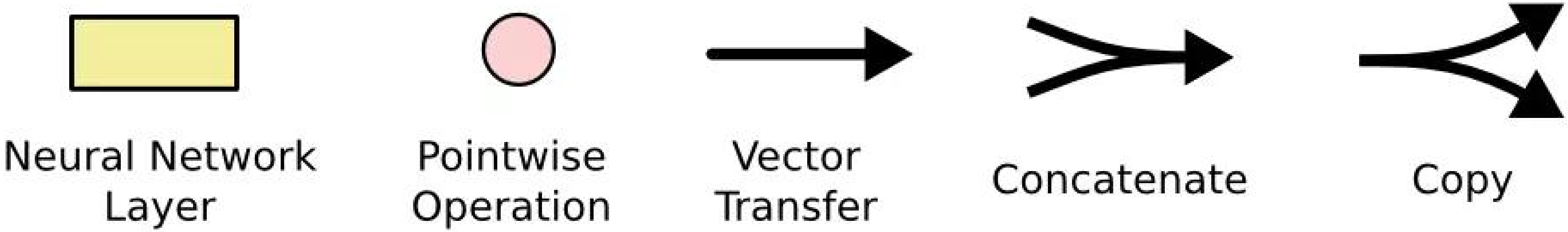
LSTM 的重复模块



除了h在随时间流动，单元状态c也在随时间流动，单元状态c就代表着长期记忆。

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

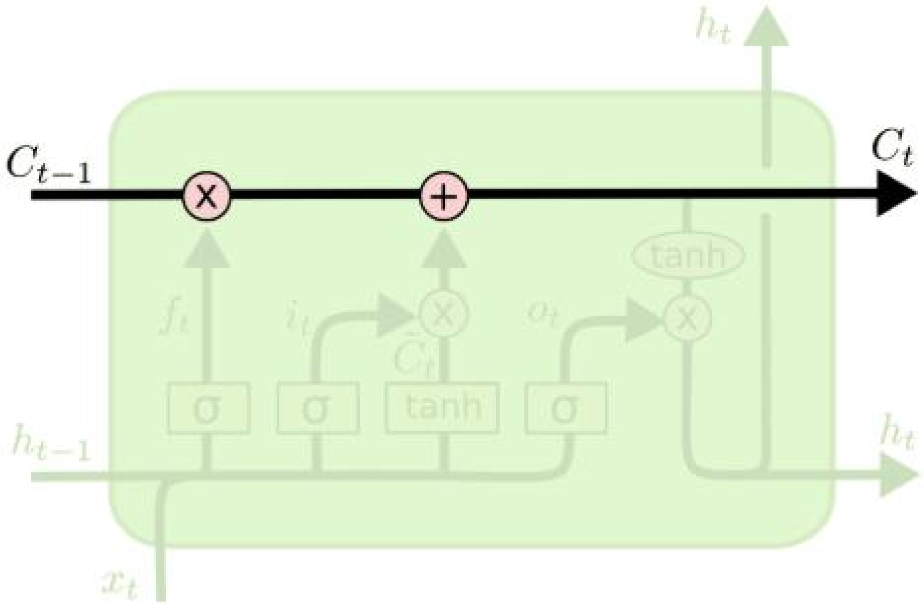
9.2 长短期记忆网络（LSTM）



- 黄色的矩形是学习得到的神经网络层
- 粉色的圆形表示运算操作，诸如加法乘法
- 黑色的单箭头表示向量传输
- 两个箭头合成一个表示向量连接
- 一个箭头分叉表示向量复制

LSTM 的核心思想

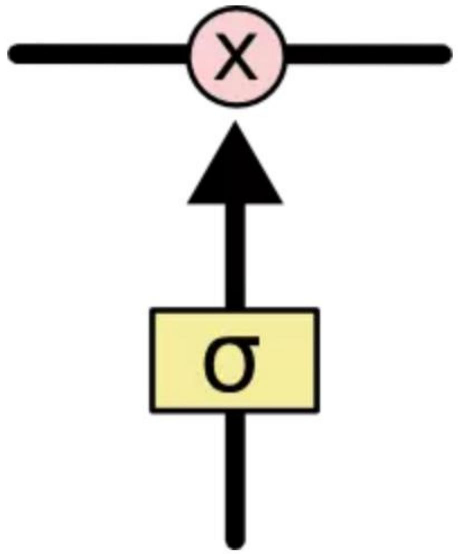
LSTM 关键是单元状态，如水平线在图上方贯穿运行。



单元状态的传递类似于传送带，其直接在整个链上运行，中间只有少量的线性交互，容易保存相关信息。

前面描述的开关如何在算法中实现呢？

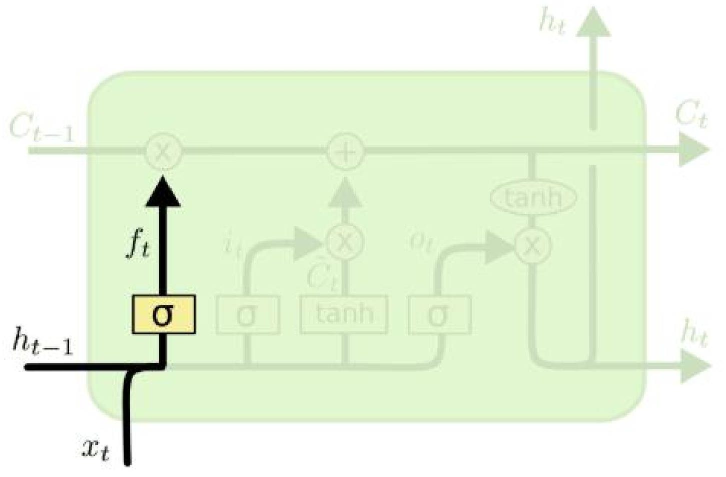
LSTM 通过精心设计的称作为“**门**”（**gate**）的结构来去除或者增加单元状态中的信息。门是一种让信息选择式通过的方法。



此门包含一个 sigmoid 神经网络层和一个 pointwise 乘法操作

- LSTM用两个门来控制单元状态c的内容
  - 一个是**遗忘门**（forget gate），决定了上一时刻的单元状态c_{t-1}有多少保留到当前时刻c_t；
  - 一个是**输入门**（input gate），决定了当前时刻网络的输入x_t有多少保存到单元状态c_t。
- LSTM用**输出门**（output gate）来控制单元状态c_t有多少输出到LSTM的当前输出值h_t

逐步理解 LSTM之遗忘门



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

权重矩阵W_f 是两个矩阵拼接而来的，

一个是 W_{fh} ，它对应输入项h_{t-1} 其维度为$d_c \times d_h$

一个是 W_{fx} ，它对应这输入项x_t ，其维度为$d_c \times d_x$

$$[W_f]\begin{bmatrix}\mathbf{h}_{t-1}\\ \mathbf{x}_t\end{bmatrix} = \begin{bmatrix}W_{fh} & W_{fx}\end{bmatrix}\begin{bmatrix}\mathbf{h}_{t-1}\\ \mathbf{x}_t\end{bmatrix} = W_{fh}\mathbf{h}_{t-1} + W_{fx}\mathbf{x}_t$$

上式中，W_f 是遗忘门的权重矩阵，

$[\mathbf{h}_{t-1}, \mathbf{x}_t]$ 表示把两个向量连接成一个更长的向量，

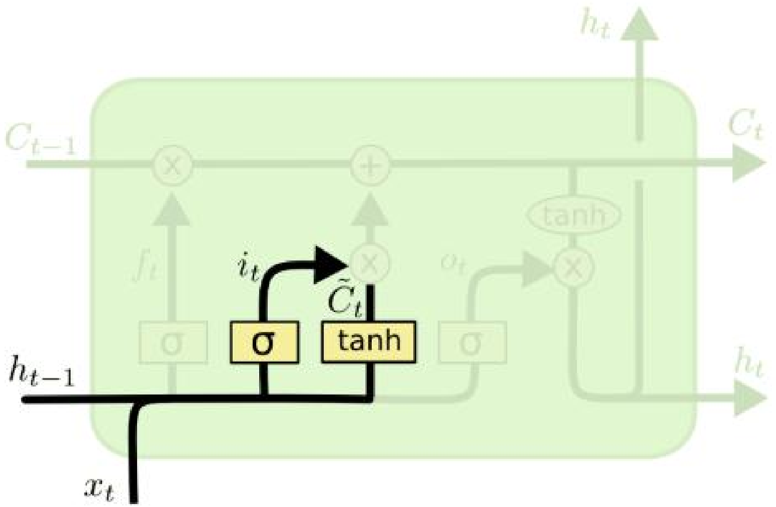
b_f 是遗忘门的偏置项，

σ 是sigmoid函数，如果输入的维度是 d_x ，隐藏层的维度是d_h ，

单元状态的维度是 d_c，则遗忘门的权重矩阵W_f

的维度是 $d_c \times (d_h + d_x)$ 。

逐步理解 LSTM之输入门



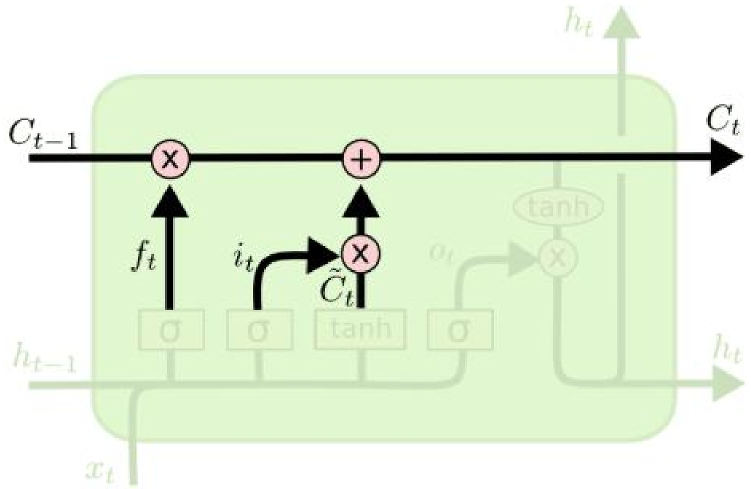
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

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

sigmoid 函数称 为输入门，决定将要更新什么值

tanh 层创建一个新的候选值向量，$\tilde{C}_t$ 会被加入到状态中

逐步理解 LSTM之更新单元状态

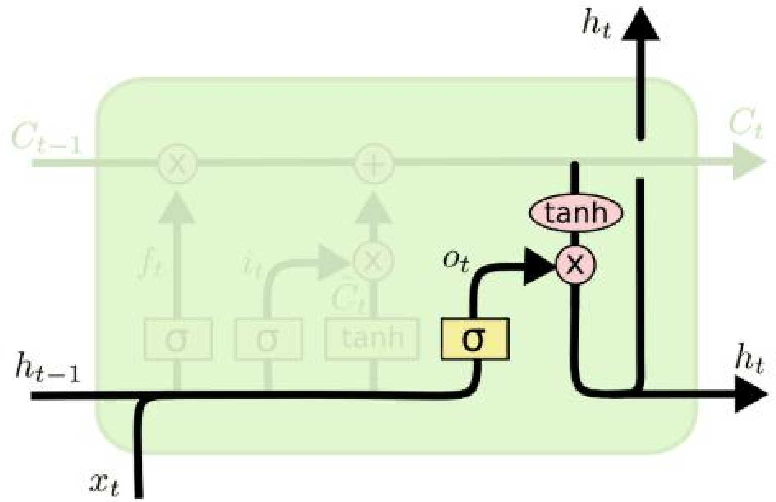


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

现在开始计算当前时刻的单元状态C_t。它是由上一次的单元状态 C_{t-1} 按原元素乘以遗忘门f_t ，再用当前输入的单元状态 $\tilde{C}_t$ 按元素乘以输入门 i_t ，再将两个积加和产生的。

由于遗忘门的控制，它可以保存很久很久之前的信息，由于输入门的控制，它又可以避免当前无关紧要的内容进入记忆。

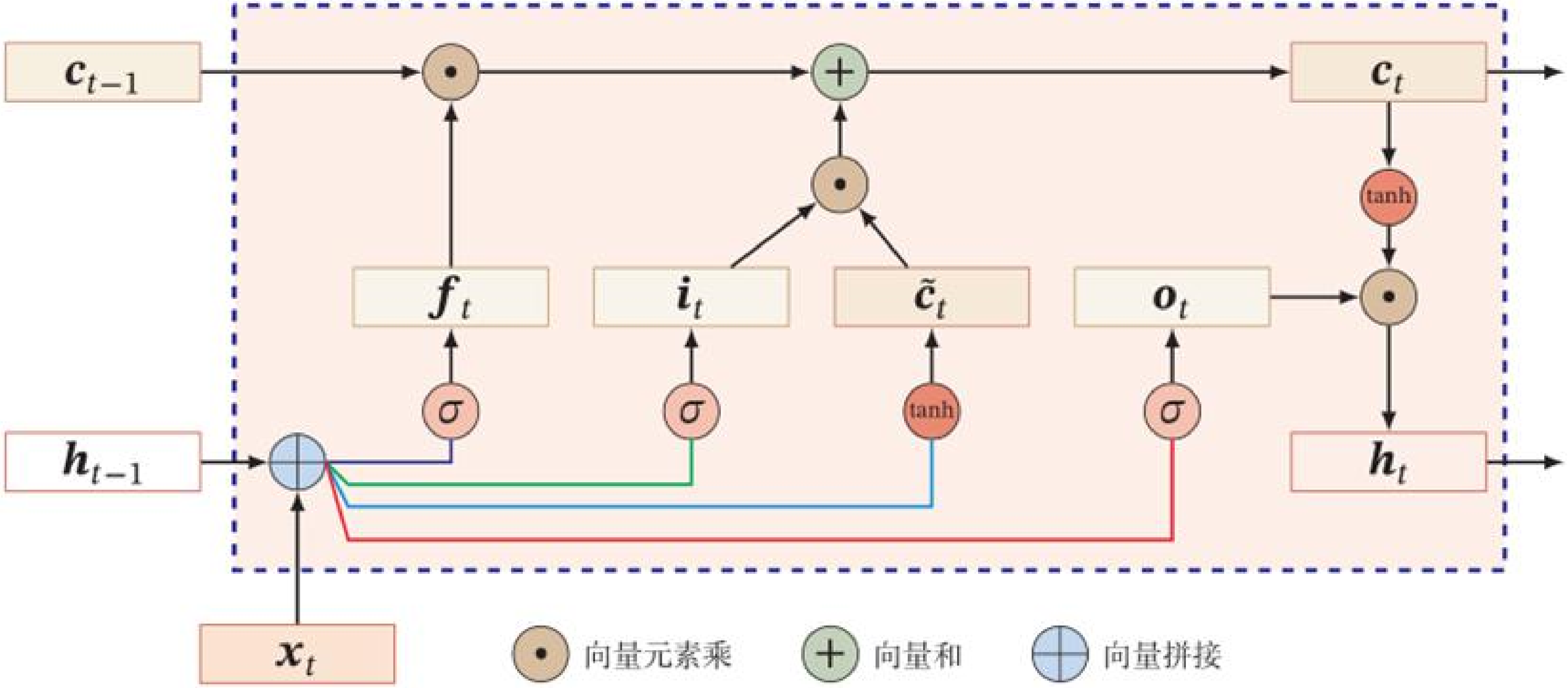
逐步理解 LSTM之输出门



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

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

输出门控制了长期记忆对当前输出的影响，由输出门和单元状态共同确定



$$\mathbf{i}_t = \sigma(W_i\mathbf{x}_t + U_i\mathbf{h}_{t-1} + \mathbf{b}_i),$$

$$\mathbf{f}_t = \sigma(W_f\mathbf{x}_t + U_f\mathbf{h}_{t-1} + \mathbf{b}_f),$$

$$\mathbf{o}_t = \sigma(W_o\mathbf{x}_t + U_o\mathbf{h}_{t-1} + \mathbf{b}_o),$$

$$\tilde{\mathbf{c}}_t = \tanh(W_c\mathbf{x}_t + U_c\mathbf{h}_{t-1} + \mathbf{b}_c)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t,$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t\right),$$

目录

9.1 长程依赖问题

9.2 长短期记忆网络（LSTM）

- 长短期记忆网络的训练

9.3 门控循环神经网络（GRU）

9.4 深层循环神经网络

□ LSTM训练算法框架

LSTM的训练算法仍然是反向传播算法。主要有下面三个步骤：

a) 前向计算每个神经元的输出值，对于LSTM来说，即 $\mathbf{f}_t, \mathbf{i}_t, \mathbf{c}_t, \mathbf{o}_t, \mathbf{h}_t$ 五向量的值。计算方法如前所述。

b) 反向计算每个神经元的**误差项**值。与**循环神经网络**一样，LSTM误差项的反向传播也是包括两个方向：一个是沿时间的反向传播，即从当前t时刻开始，计算每个时刻的误差项；一个是将误差项向上一层传播。

c) 根据相应的误差项，计算每个权重的梯度。

关于公式和符号的说明

接下来的推导中，我们设定门gate的激活函数为sigmoid函数，输出的激活函数为tanh函数。他们的导数分别为：

$$\sigma(z) = y = \frac{1}{1+e^{-z}}$$

$$\sigma'(z) = y(1-y)$$

$$\tanh(z) = y = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

$$\tanh'(z) = 1 - y^2$$

sigmoid和tanh函数的导数都是原函数的函数。这样，我们一旦计算原函数的值，就可以用它来计算出导数的值。

LSTM需要学习的参数共有8组，分别是：遗忘门的权重矩阵 W_f 和偏置项 b_f 、输入门的权重矩阵 W_i 和偏置项 b_i 、输出门的权重矩阵 W_o 和偏置项 b_o ，以及计算单元状态的权重矩阵 W_c 和偏置项 。因为权重矩阵的两部分在反向传播中使用不同的公式，因此在后续的推导中，权重矩阵 W_f W_i W_o W_c 都将被写为分开的两个矩阵 W_{fh} W_{fx} W_{ih} W_{ix} W_{oh} W_{ox} W_{ch} W_{cx}

○表示按元素乘 。**两向量元素乘计算**如下：

$$\mathbf{a} \circ \mathbf{b} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \dots \\ a_n \end{bmatrix} \circ \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \dots \\ b_n \end{bmatrix} = \begin{bmatrix} a_1 b_1 \\ a_2 b_2 \\ a_3 b_3 \\ \dots \\ a_n b_n \end{bmatrix}$$

当○作用于一个**向量**和一个**矩阵**时：

$$\mathbf{a} \circ X = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \dots \\ a_n \end{bmatrix} \circ \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1n} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2n} \\ x_{31} & x_{32} & x_{33} & \dots & x_{3n} \\ & & \dots & & \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{nn} \end{bmatrix}$$

当 ○ 作用于两个**矩阵**时，两个矩阵对应位置的元素相乘。按元素乘可以在某些情况下简化矩阵和向量运算。例如，当一个对角矩阵右乘一个矩阵时，相当于用对角矩阵的对角线组成的向量按元素乘那个矩阵：

$$\mathrm{diag}[\mathbf{a}]X = \mathbf{a} \circ X$$

当一个行向量右乘一个对角矩阵时，相当于这个行向量按元素乘那个矩阵对角线组成的向量：

$$\mathbf{a}^{\mathrm{T}} diag[\mathbf{b}] = \mathbf{a} \circ \mathbf{b}$$

上面这两点，在我们后续推导中会多次用到。

9.2 长短期记忆网络（LSTM）

在t时刻，LSTM的输出值为 $\mathbf{h}_t$ 。我们定义t时刻的误差项 δ_t 为 ： $\delta_t \overset{\text{def}}{=} \frac{\partial E}{\partial \mathbf{h}_t}$

LSTM有四个加权输入，分别对应 $\mathbf{f}_t, \mathbf{i}_t, \mathbf{c}_t, \mathbf{o}_t$ ，我们希望往上一层传递一个误差项而不是四个，但我们仍然需要定义四个加权输入，以及他们对应的误差项。

$$\begin{aligned}
\mathbf{net}_{f,t} &= W_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f \\
&= W_{fh}\mathbf{h}_{t-1} + W_{fx}\mathbf{x}_t + \mathbf{b}_f \\
\mathbf{net}_{i,t} &= W_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i \\
&= W_{ih}\mathbf{h}_{t-1} + W_{ix}\mathbf{x}_t + \mathbf{b}_i \\
\mathbf{net}_{\tilde{c},t} &= W_c[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c \\
&= W_{ch}\mathbf{h}_{t-1} + W_{cx}\mathbf{x}_t + \mathbf{b}_c \\
\mathbf{net}_{o,t} &= W_o[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o \\
&= W_{oh}\mathbf{h}_{t-1} + W_{ox}\mathbf{x}_t + \mathbf{b}_o
\end{aligned}$$

$$\delta_{f,t} \overset{\text{def}}{=} \frac{\partial E}{\partial \mathbf{net}_{f,t}}$$

$$\delta_{i,t} \overset{\text{def}}{=} \frac{\partial E}{\partial \mathbf{net}_{i,t}}$$

$$\delta_{\tilde{c},t} \overset{\text{def}}{=} \frac{\partial E}{\partial \mathbf{net}_{\tilde{c},t}}$$

$$\delta_{o,t} \overset{\text{def}}{=} \frac{\partial E}{\partial \mathbf{net}_{o,t}}$$

误差项沿时间的反向传递

沿时间反向传递误差项，就是要计算出t-1时刻的误差项 δ_{t-1} 。

$$\begin{aligned}\delta_{t-1}^{\mathrm{T}} &= \frac{\partial E}{\partial \mathbf{h_{t-1}}} \\ &= \frac{\partial E}{\partial \mathbf{h_t}} \frac{\partial \mathbf{h_t}}{\partial \mathbf{h_{t-1}}} \\ &= \delta_t^{\mathrm{T}} \frac{\partial \mathbf{h_t}}{\partial \mathbf{h_{t-1}}}\end{aligned}$$

我们知道，$\frac{\partial \mathbf{h_t}}{\partial \mathbf{h_{t-1}}}$ 是一个Jacobian矩阵。如果隐藏层h的维度是N的话，那么它就是一个$N \times N$矩阵。为了求出它，可利用$\mathbf{h}_t$ 的公式：

$$\mathbf{h}_t = \mathbf{o}_t \circ \tanh(\mathbf{c}_t)$$
$$\mathbf{c}_t = \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tilde{\mathbf{c}}_t$$

利用全导数公式可得：

$$\begin{aligned}\delta_t^{\mathrm{T}} \frac{\partial \mathbf{h_t}}{\partial \mathbf{h_{t-1}}} &= \delta_t^{\mathrm{T}} \frac{\partial \mathbf{h_t}}{\partial \mathbf{o}_t} \frac{\partial \mathbf{o}_t}{\partial \mathbf{net}_{o,t}} \frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h_{t-1}}} + \delta_t^{\mathrm{T}} \frac{\partial \mathbf{h_t}}{\partial \mathbf{c}_t} \frac{\partial \mathbf{c}_t}{\partial \mathbf{f_t}} \frac{\partial \mathbf{f_t}}{\partial \mathbf{net}_{f,t}} \frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h_{t-1}}} + \delta_t^{\mathrm{T}} \frac{\partial \mathbf{h_t}}{\partial \mathbf{c}_t} \frac{\partial \mathbf{c}_t}{\partial \mathbf{i_t}} \frac{\partial \mathbf{i_t}}{\partial \mathbf{net}_{i,t}} \frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h_{t-1}}} + \delta_t^{\mathrm{T}} \frac{\partial \mathbf{h_t}}{\partial \mathbf{c}_t} \frac{\partial \mathbf{c}_t}{\partial \tilde{\mathbf{c}}_t} \frac{\partial \tilde{\mathbf{c}}_t}{\partial \mathbf{net}_{\tilde{c},t}} \frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{h_{t-1}}} \\ &= \delta_{o,t}^{\mathrm{T}} \frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h_{t-1}}} + \delta_{f,t}^{\mathrm{T}} \frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h_{t-1}}} + \delta_{i,t}^{\mathrm{T}} \frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h_{t-1}}} + \delta_{\tilde{c},t}^{\mathrm{T}} \frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{h_{t-1}}} \qquad \text{(式 7)}\end{aligned}$$

下面，我们要把**式7**中的每个偏导数都求出来。根据下式，我们可以求出:

$$\frac{\partial \mathbf{h_t}}{\partial \mathbf{o}_t} = \mathrm{diag}[\tanh(\mathbf{c}_t)]$$

$$\frac{\partial \mathbf{h_t}}{\partial \mathbf{c}_t} = \mathrm{diag}\left[\mathbf{o}_t \circ \left(1 - \tanh\left(\mathbf{c}_t\right)^2\right)\right]$$

$$\frac{\partial \mathbf{c}_t}{\partial \mathbf{f_t}} = \mathrm{diag}[\mathbf{c}_{t-1}]$$

$$\frac{\partial \mathbf{c}_t}{\partial \mathbf{i_t}} = \mathrm{diag}[\tilde{\mathbf{c}}_t]$$

$$\frac{\partial \mathbf{c}_t}{\partial \tilde{\mathbf{c}}_\mathrm{t}} = \mathrm{diag}[\mathbf{i}_t]$$

$$\mathbf{i}_t = \sigma(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1} + \mathbf{b}_i),$$

$$\mathbf{f}_t = \sigma(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + \mathbf{b}_f),$$

$$\mathbf{o}_t = \sigma(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1} + \mathbf{b}_o),$$

$$\tilde{\mathbf{c}}_t = \tanh(W_c \mathbf{x}_t + U_c \mathbf{h}_{t-1} + \mathbf{b}_c)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t,$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t\right),$$

9.2 长短期记忆网络（LSTM）

因为：

$$
\begin{aligned}
\mathbf{o}_t &= \sigma(\mathbf{net}_{o,t}) \\
\mathbf{net}_{o,t} &= W_{oh}\mathbf{h}_{t-1} + W_{ox}\mathbf{x}_t + \mathbf{b}_o \\
\mathbf{f}_t &= \sigma(\mathbf{net}_{f,t}) \\
\mathbf{net}_{f,t} &= W_{fh}\mathbf{h}_{t-1} + W_{fx}\mathbf{x}_t + \mathbf{b}_f \\
\mathbf{i}_t &= \sigma(\mathbf{net}_{i,t}) \\
\mathbf{net}_{i,t} &= W_{ih}\mathbf{h}_{t-1} + W_{ix}\mathbf{x}_t + \mathbf{b}_i \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{net}_{\tilde{c},t}) \\
\mathbf{net}_{\tilde{c},t} &= W_{ch}\mathbf{h}_{t-1} + W_{cx}\mathbf{x}_t + \mathbf{b}_c
\end{aligned}
\quad\Longrightarrow\quad
\begin{aligned}
\frac{\partial \mathbf{o}_t}{\partial \mathbf{net}_{o,t}} &= \mathrm{diag}[\mathbf{o}_t \circ (1-\mathbf{o}_t)] \\
\frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h_{t-1}}} &= W_{oh} \\
\frac{\partial \mathbf{f}_t}{\partial \mathbf{net}_{f,t}} &= \mathrm{diag}[\mathbf{f}_t \circ (1-\mathbf{f}_t)] \\
\frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h}_{t-1}} &= W_{fh} \\
\frac{\partial \mathbf{i}_t}{\partial \mathbf{net}_{i,t}} &= \mathrm{diag}[\mathbf{i}_t \circ (1-\mathbf{i}_t)] \\
\frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h}_{t-1}} &= W_{ih} \\
\frac{\partial \tilde{\mathbf{c}}_t}{\partial \mathbf{net}_{\tilde{c},t}} &= \mathrm{diag}\left[1-\tilde{\mathbf{c}}_t^2\right] \\
\frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{h}_{t-1}} &= W_{ch}
\end{aligned}
$$

将上述偏导数带入到式7，得到：

$$\delta_t^{\mathrm{T}}\frac{\partial \mathbf{h_t}}{\partial \mathbf{h_{t-1}}}=\delta_t^{\mathrm{T}}\frac{\partial \mathbf{h_t}}{\partial \mathbf{o}_t}\frac{\partial \mathbf{o}_t}{\partial \mathbf{net}_{o,t}}\frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h_{t-1}}}+\delta_t^{\mathrm{T}}\frac{\partial \mathbf{h_t}}{\partial \mathbf{c}_t}\frac{\partial \mathbf{c}_t}{\partial \mathbf{f_t}}\frac{\partial \mathbf{f}_t}{\partial \mathbf{net}_{f,t}}\frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h_{t-1}}}+\delta_t^{\mathrm{T}}\frac{\partial \mathbf{h_t}}{\partial \mathbf{c}_t}\frac{\partial \mathbf{c}_t}{\partial \mathbf{i_t}}\frac{\partial \mathbf{i}_t}{\partial \mathbf{net}_{i,t}}\frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h_{t-1}}}+\delta_t^{\mathrm{T}}\frac{\partial \mathbf{h_t}}{\partial \mathbf{c}_t}\frac{\partial \mathbf{c}_t}{\partial \tilde{\mathbf{c}}_t}\frac{\partial \tilde{\mathbf{c}}_t}{\partial \mathbf{net}_{\tilde{c},t}}\frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{h_{t-1}}}$$

$$=\delta_{o,t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h_{t-1}}}+\delta_{f,t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h_{t-1}}}+\delta_{i,t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h_{t-1}}}+\delta_{\tilde{c},t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{h_{t-1}}} \qquad (\text{式 } 7)$$

$$\delta_{t-1}=\delta_{o,t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h_{t-1}}}+\delta_{f,t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h_{t-1}}}+\delta_{i,t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h_{t-1}}}+\delta_{\tilde{c},t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{h_{t-1}}}$$

$$=\delta_{o,t}^{\mathrm{T}}W_{oh}+\delta_{f,t}^{\mathrm{T}}W_{fh}+\delta_{i,t}^{\mathrm{T}}W_{ih}+\delta_{\tilde{c},t}^{\mathrm{T}}W_{ch} \qquad (\text{式 } 8)$$

$$
\begin{aligned}
\delta_{t-1} &= \delta_{o,t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h_{t-1}}} + \delta_{f,t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h_{t-1}}} + \delta_{i,t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h_{t-1}}} + \delta_{\tilde{c},t}^{\mathrm{T}}\frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{h_{t-1}}} \\
&= \delta_{o,t}^{\mathrm{T}} W_{oh} + \delta_{f,t}^{\mathrm{T}} W_{fh} + \delta_{i,t}^{\mathrm{T}} W_{ih} + \delta_{\tilde{c},t}^{\mathrm{T}} W_{ch} \qquad (\text{式 } 8)
\end{aligned}
$$

根据 $\delta_{o,t}$ $\delta_{f,t}$ $\delta_{i,t}$ $\delta_{\tilde{c},t}$ 的定义，可知：

$$
\begin{aligned}
\delta_{o,t}^{\mathrm{T}} &= \delta_t^{\mathrm{T}} \circ \tanh(\mathbf{c}_t) \circ \mathbf{o}_t \circ (1 - \mathbf{o}_t) && (\text{式 } 9) \\
\delta_{f,t}^{\mathrm{T}} &= \delta_t^{\mathrm{T}} \circ \mathbf{o}_t \circ (1 - \tanh(\mathbf{c}_t)^2) \circ \mathbf{c}_{t-1} \circ \mathbf{f}_t \circ (1 - \mathbf{f}_t) && (\text{式 } 10) \\
\delta_{i,t}^{\mathrm{T}} &= \delta_t^{\mathrm{T}} \circ \mathbf{o}_t \circ (1 - \tanh(\mathbf{c}_t)^2) \circ \tilde{\mathbf{c}}_t \circ \mathbf{i}_t \circ (1 - \mathbf{i}_t) && (\text{式 } 11) \\
\delta_{\tilde{c},t}^{\mathrm{T}} &= \delta_t^{\mathrm{T}} \circ \mathbf{o}_t \circ (1 - \tanh(\mathbf{c}_t)^2) \circ \mathbf{i}_t \circ (1 - \tilde{\mathbf{c}}^2) && (\text{式 } 12)
\end{aligned}
$$

式8到式12就是将误差沿时间反向传播一个时刻的公式。基于这些公式，我们可以写出将误差项向前传递到任意时刻。

将误差项传递到上一层

我们假设当前为第l层，定义l-1层的误差项是误差函数对l-1层**加权输入**的导数，即：

$$\delta_t^{l-1} \stackrel{\text{def}}{=} \frac{\partial E}{\mathbf{net}_t^{l-1}}$$

本层LSTM的输入x_t 由下面的公式计算：

$$\mathbf{x}_t^l = f^{l-1}(\mathbf{net}_t^{l-1})$$

上式中f^{l-1} 表示第l-1层的**激活函数**。

9.2 长短期记忆网络（LSTM）

因为 $\mathbf{net}^l_{f,t}, \mathbf{net}^l_{i,t}, \mathbf{net}^l_{\tilde{c},t}, \mathbf{net}^l_{o,t}$ 都是 x_t 的函数，x_t 又是 net_t^{l-1} 的函数，因此，要求出E对 net_t^{l-1} 的导数，就需要使用全导数公式：

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{net}_t^{l-1}} &= \frac{\partial E}{\partial \mathbf{net}^{\mathbf{l}}_{\mathbf{f,t}}} \frac{\partial \mathbf{net}^{\mathbf{l}}_{\mathbf{f,t}}}{\partial \mathbf{x}_t^l} \frac{\partial \mathbf{x}_t^l}{\partial \mathbf{net}_{\mathbf{t}}^{\mathbf{l-1}}} + \frac{\partial E}{\partial \mathbf{net}^{\mathbf{l}}_{\mathbf{i,t}}} \frac{\partial \mathbf{net}^{\mathbf{l}}_{\mathbf{i,t}}}{\partial \mathbf{x}_t^l} \frac{\partial \mathbf{x}_t^l}{\partial \mathbf{net}_{\mathbf{t}}^{\mathbf{l-1}}} + \frac{\partial E}{\partial \mathbf{net}^{\mathbf{l}}_{\tilde{\mathbf{c}},\mathbf{t}}} \frac{\partial \mathbf{net}^{\mathbf{l}}_{\tilde{\mathbf{c}},\mathbf{t}}}{\partial \mathbf{x}_t^l} \frac{\partial \mathbf{x}_t^l}{\partial \mathbf{net}_{\mathbf{t}}^{\mathbf{l-1}}} + \frac{\partial E}{\partial \mathbf{net}^{\mathbf{l}}_{\mathbf{o,t}}} \frac{\partial \mathbf{net}^{\mathbf{l}}_{\mathbf{o,t}}}{\partial \mathbf{x}_t^l} \frac{\partial \mathbf{x}_t^l}{\partial \mathbf{net}_{\mathbf{t}}^{\mathbf{l-1}}} \\
&= \delta_{f,t}^{\mathrm{T}} W_{fx} \circ f'(\mathbf{net}_t^{l-1}) + \delta_{i,t}^{\mathrm{T}} W_{ix} \circ f'(\mathbf{net}_t^{l-1}) + \delta_{\tilde{c},t}^{\mathrm{T}} W_{cx} \circ f'(\mathbf{net}_t^{l-1}) + \delta_{o,t}^{\mathrm{T}} W_{ox} \circ f'(\mathbf{net}_t^{l-1}) \\
&= (\delta_{f,t}^{\mathrm{T}} W_{fx} + \delta_{i,t}^{\mathrm{T}} W_{ix} + \delta_{\tilde{c},t}^{\mathrm{T}} W_{cx} + \delta_{o,t}^{\mathrm{T}} W_{ox}) \circ f'(\mathbf{net}_t^{l-1}) \qquad (\text{式 } 14)
\end{aligned}
$$

式14就是将误差传递到上一层的公式。

$$
\begin{aligned}
\mathbf{o}_t &= \sigma(\mathbf{net}_{o,t}) \\
\mathbf{net}_{o,t} &= W_{oh}\mathbf{h}_{t-1} + W_{ox}\mathbf{x}_t + \mathbf{b}_o \\
\mathbf{f}_t &= \sigma(\mathbf{net}_{f,t}) \\
\mathbf{net}_{f,t} &= W_{fh}\mathbf{h}_{t-1} + W_{fx}\mathbf{x}_t + \mathbf{b}_f \\
\mathbf{i}_t &= \sigma(\mathbf{net}_{i,t}) \\
\mathbf{net}_{i,t} &= W_{ih}\mathbf{h}_{t-1} + W_{ix}\mathbf{x}_t + \mathbf{b}_i \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{net}_{\tilde{c},t}) \\
\mathbf{net}_{\tilde{c},t} &= W_{ch}\mathbf{h}_{t-1} + W_{cx}\mathbf{x}_t + \mathbf{b}_c
\end{aligned}
$$

□ 权重梯度的计算

对于权重

$$W_{fh}, W_{ih}, W_{ch}, W_{oh}$$

梯度，我们知道它的梯度是各个时刻梯度之和，我们首先求出它们在t时刻的梯度，然后再求出他们最终的梯度。

$$\frac{\partial E}{\partial W_{oh,t}} = \frac{\partial E}{\partial \mathbf{net}_{o,t}} \frac{\partial \mathbf{net}_{o,t}}{\partial W_{oh,t}} = \delta_{o,t} \mathbf{h}_{t-1}^{\mathrm{T}}$$

$$\frac{\partial E}{\partial W_{fh,t}} = \frac{\partial E}{\partial \mathbf{net}_{f,t}} \frac{\partial \mathbf{net}_{f,t}}{\partial W_{fh,t}} = \delta_{f,t} \mathbf{h}_{t-1}^{\mathrm{T}}$$

$$\frac{\partial E}{\partial W_{ih,t}} = \frac{\partial E}{\partial \mathbf{net}_{i,t}} \frac{\partial \mathbf{net}_{i,t}}{\partial W_{ih,t}} = \delta_{i,t} \mathbf{h}_{t-1}^{\mathrm{T}}$$

$$\frac{\partial E}{\partial W_{ch,t}} = \frac{\partial E}{\partial \mathbf{net}_{\tilde{c},t}} \frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial W_{ch,t}} = \delta_{\tilde{c},t} \mathbf{h}_{t-1}^{\mathrm{T}}$$

将各个时刻的梯度加在一起，就能得到最终的梯度：

$$\frac{\partial E}{\partial W_{oh}} = \sum_{j=1}^{t} \delta_{o,j} \mathbf{h}_{j-1}^{\mathrm{T}}$$

$$\frac{\partial E}{\partial W_{fh}} = \sum_{j=1}^{t} \delta_{f,j} \mathbf{h}_{j-1}^{\mathrm{T}}$$

$$\frac{\partial E}{\partial W_{ih}} = \sum_{j=1}^{t} \delta_{i,j} \mathbf{h}_{j-1}^{\mathrm{T}}$$

$$\frac{\partial E}{\partial W_{ch}} = \sum_{j=1}^{t} \delta_{\tilde{c},j} \mathbf{h}_{j-1}^{\mathrm{T}}$$

$$\begin{aligned}
\mathbf{o}_t &= \sigma(\mathbf{net}_{o,t}) \\
\mathbf{net}_{o,t} &= W_{oh}\mathbf{h}_{t-1} + W_{ox}\mathbf{x}_t + \mathbf{b}_o \\
\mathbf{f}_t &= \sigma(\mathbf{net}_{f,t}) \\
\mathbf{net}_{f,t} &= W_{fh}\mathbf{h}_{t-1} + W_{fx}\mathbf{x}_t + \mathbf{b}_f \\
\mathbf{i}_t &= \sigma(\mathbf{net}_{i,t}) \\
\mathbf{net}_{i,t} &= W_{ih}\mathbf{h}_{t-1} + W_{ix}\mathbf{x}_t + \mathbf{b}_i \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{net}_{\tilde{c},t}) \\
\mathbf{net}_{\tilde{c},t} &= W_{ch}\mathbf{h}_{t-1} + W_{cx}\mathbf{x}_t + \mathbf{b}_c
\end{aligned}$$

对于偏置项 $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_c, \mathbf{b}_o$ 的梯度，也是将各个时刻的梯度加在一起。下面是各个时刻的偏置项梯度：

$$\begin{aligned}\frac{\partial E}{\partial \mathbf{b}_{o,t}} &= \frac{\partial E}{\partial \mathbf{net}_{o,t}} \frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{b}_{o,t}} \\ &= \delta_{o,t} \\ \frac{\partial E}{\partial \mathbf{b}_{f,t}} &= \frac{\partial E}{\partial \mathbf{net}_{f,t}} \frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{b}_{f,t}} \\ &= \delta_{f,t} \\ \frac{\partial E}{\partial \mathbf{b}_{i,t}} &= \frac{\partial E}{\partial \mathbf{net}_{i,t}} \frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{b}_{i,t}} \\ &= \delta_{i,t} \\ \frac{\partial E}{\partial \mathbf{b}_{c,t}} &= \frac{\partial E}{\partial \mathbf{net}_{\tilde{c},t}} \frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{b}_{c,t}} \\ &= \delta_{\tilde{c},t}\end{aligned}$$

下面是最终的偏置项梯度，即将各个时刻的偏置项梯度加在一起：

$$\begin{aligned}\frac{\partial E}{\partial \mathbf{b}_o} &= \sum_{j=1}^{t} \delta_{o,j} \\ \frac{\partial E}{\partial \mathbf{b}_i} &= \sum_{j=1}^{t} \delta_{i,j} \\ \frac{\partial E}{\partial \mathbf{b}_f} &= \sum_{j=1}^{t} \delta_{f,j} \\ \frac{\partial E}{\partial \mathbf{b}_c} &= \sum_{j=1}^{t} \delta_{\tilde{c},j}\end{aligned}$$

$$\begin{aligned}\mathbf{o}_t &= \sigma(\mathbf{net}_{o,t}) \\ \mathbf{net}_{o,t} &= W_{oh}\mathbf{h}_{t-1} + W_{ox}\mathbf{x}_t + \mathbf{b}_o \\ \mathbf{f}_t &= \sigma(\mathbf{net}_{f,t}) \\ \mathbf{net}_{f,t} &= W_{fh}\mathbf{h}_{t-1} + W_{fx}\mathbf{x}_t + \mathbf{b}_f \\ \mathbf{i}_t &= \sigma(\mathbf{net}_{i,t}) \\ \mathbf{net}_{i,t} &= W_{ih}\mathbf{h}_{t-1} + W_{ix}\mathbf{x}_t + \mathbf{b}_i \\ \tilde{\mathbf{c}}_t &= \tanh(\mathbf{net}_{\tilde{c},t}) \\ \mathbf{net}_{\tilde{c},t} &= W_{ch}\mathbf{h}_{t-1} + W_{cx}\mathbf{x}_t + \mathbf{b}_c\end{aligned}$$

对于 $W_{fx}, W_{ix}, W_{cx}, W_{ox}$ 的梯度，只需要根据相应的误差项直接计算即可

$$\begin{aligned}\frac{\partial E}{\partial W_{ox}} &= \frac{\partial E}{\partial \mathbf{net}_{o,t}} \frac{\partial \mathbf{net}_{o,t}}{\partial W_{ox}} \\ &= \delta_{o,t} \mathbf{x}_t^{\mathrm{T}}\end{aligned}$$

$$\begin{aligned}\frac{\partial E}{\partial W_{fx}} &= \frac{\partial E}{\partial \mathbf{net}_{f,t}} \frac{\partial \mathbf{net}_{f,t}}{\partial W_{fx}} \\ &= \delta_{f,t} \mathbf{x}_t^{\mathrm{T}}\end{aligned}$$

$$\begin{aligned}\frac{\partial E}{\partial W_{ix}} &= \frac{\partial E}{\partial \mathbf{net}_{i,t}} \frac{\partial \mathbf{net}_{i,t}}{\partial W_{ix}} \\ &= \delta_{i,t} \mathbf{x}_t^{\mathrm{T}}\end{aligned}$$

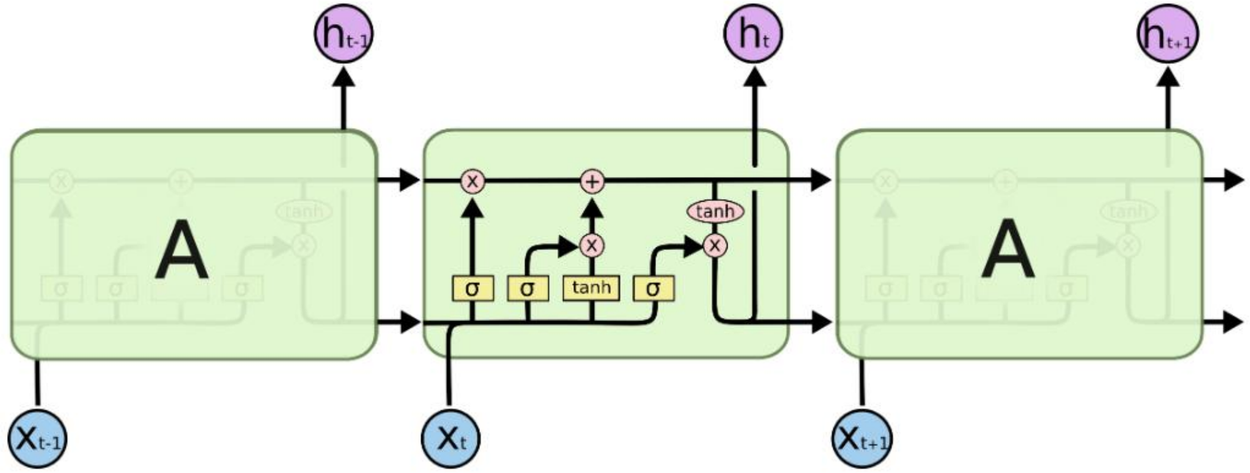
$$\begin{aligned}\frac{\partial E}{\partial W_{cx}} &= \frac{\partial E}{\partial \mathbf{net}_{\tilde{c},t}} \frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial W_{cx}} \\ &= \delta_{\tilde{c},t} \mathbf{x}_t^{\mathrm{T}}\end{aligned}$$

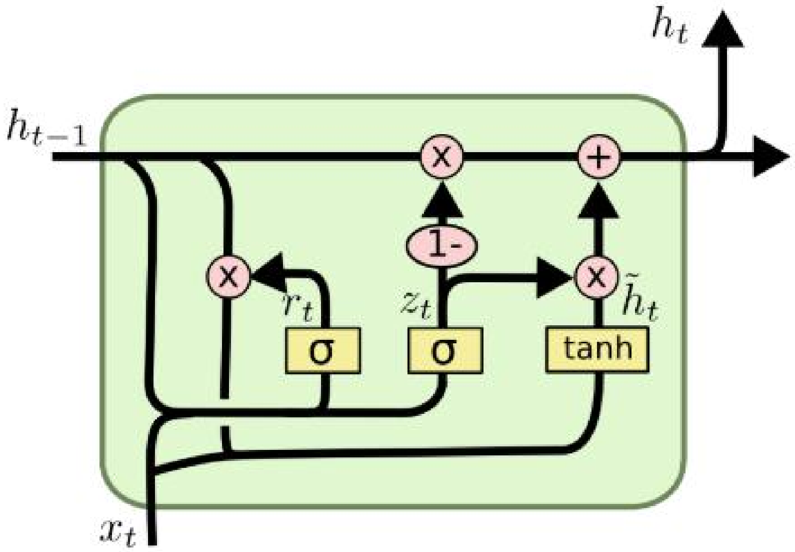
$$\begin{aligned}\mathbf{o}_t &= \sigma(\mathbf{net}_{o,t}) \\ \mathbf{net}_{o,t} &= W_{oh}\mathbf{h}_{t-1} + W_{ox}\mathbf{x}_t + \mathbf{b}_o \\ \mathbf{f}_t &= \sigma(\mathbf{net}_{f,t}) \\ \mathbf{net}_{f,t} &= W_{fh}\mathbf{h}_{t-1} + W_{fx}\mathbf{x}_t + \mathbf{b}_f \\ \mathbf{i}_t &= \sigma(\mathbf{net}_{i,t}) \\ \mathbf{net}_{i,t} &= W_{ih}\mathbf{h}_{t-1} + W_{ix}\mathbf{x}_t + \mathbf{b}_i \\ \tilde{\mathbf{c}}_t &= \tanh(\mathbf{net}_{\tilde{c},t}) \\ \mathbf{net}_{\tilde{c},t} &= W_{ch}\mathbf{h}_{t-1} + W_{cx}\mathbf{x}_t + \mathbf{b}_c\end{aligned}$$

目录

9.3 门控循环神经网络（GRU）

GRU（Gate Recurrent Unit）是一种循环神经网络。类似于LSTM，其主要目的也是为了解决长期记忆和反向传播中的梯度等问题。



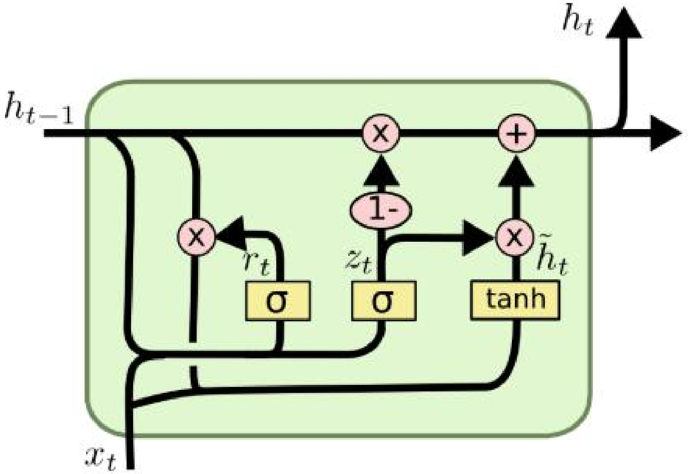


LSTM引入了三个门函数：输入门、遗忘门和输出门来控制输入值、记忆值和输出值；

GRU模型中只有两个门：分别是更新门和重置门（z_t和r_t）；

GRU是LSTM的一种变体，相较于LSTM网络，其结构更加简单，且效果很好。

9.3 门控循环神经网络（GRU）



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

图中的z_t和r_t分别表示更新门和重置门。

更新门用于控制前一时刻的状态信息被带入到当前状态中的程度，当z_t为0时，则保持h_t=h_{t-1}，即记忆前一步的值。当z_t为1时，则将h_t更新为h_t～。

重置门控制前一状态有多少信息被写入到当前的候选集h_t～上，重置门越小，前一状态的信息被写入的越少。

https://www.cnblogs.com/jiangxinyang/p/9376021.html

LSTM与GRU

- GRU的参数更少，因而训练稍快或需要更少的数据以确保它的泛化能力。
- 如果有足够的数据，LSTM的强大表达能力可能会产生更好的结果。

Greff, et al. (2016)对流行的LSTM变种做了对比实验，发现它们的表现几乎一致。

Jozefowicz, et al. (2015)测试了超过一万中RNN结构，发现某些任务情形下，有些变种比LSTM工作得更好。

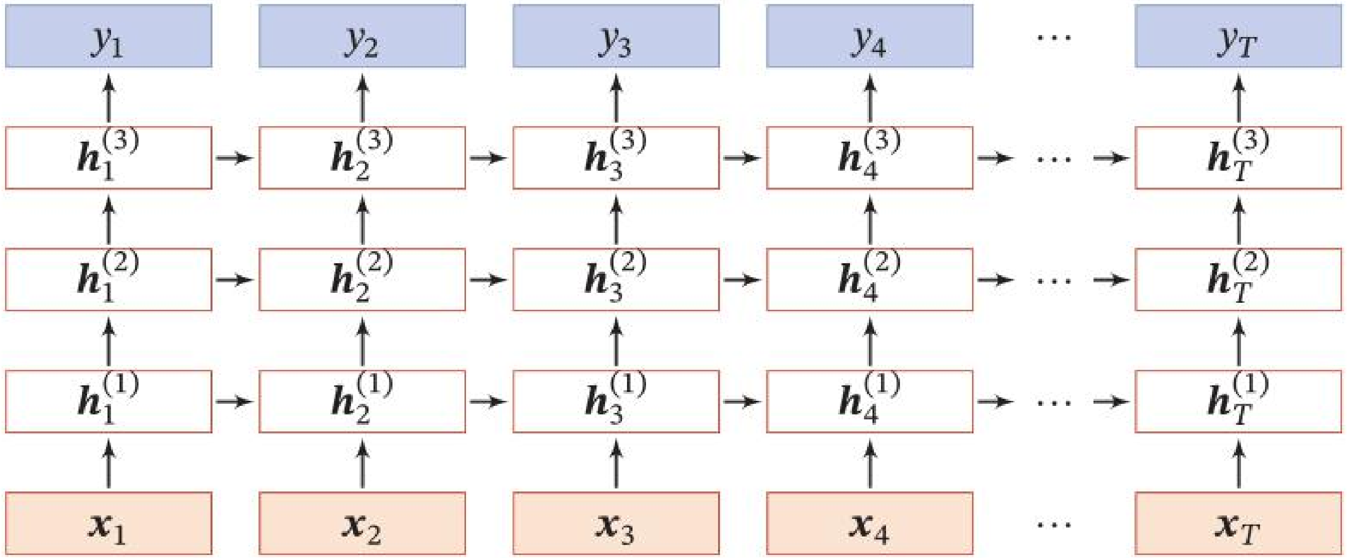
Greff K, Srivastava R K, Koutník J, et al. LSTM: A search space odyssey[J]. IEEE transactions on neural networks and learning systems, 2016, 28(10): 2222-2232.
Jozefowicz R, Zaremba W, Sutskever I. An empirical exploration of recurrent network architectures[C]//International conference on machine learning. 2015: 2342-2350.

目录

9.4 深层循环神经网络

- 循环神经网络是可深可浅的网络
  - 深网络：把循环网络按时间展开，长时间间隔的状态之间的路径很长
  - 浅网络：同一时刻网络输入到输出之间的路径 $x_t \rightarrow y_t$ 非常浅
- 增加循环神经网络的深度
  - 增强循环神经网络的能力
  - 增加同一时刻网络输入到输出之间的路径 $x_t \rightarrow y_t$，如增加隐状态到输出 $h_t \rightarrow y_t$ ，以及输入到隐状态 $x_t \rightarrow h_t$之间的路径的深度

堆叠循环神经网络(Stacked Recurrent Neural Network, SRNN)
即把多个循环网络堆叠起来。

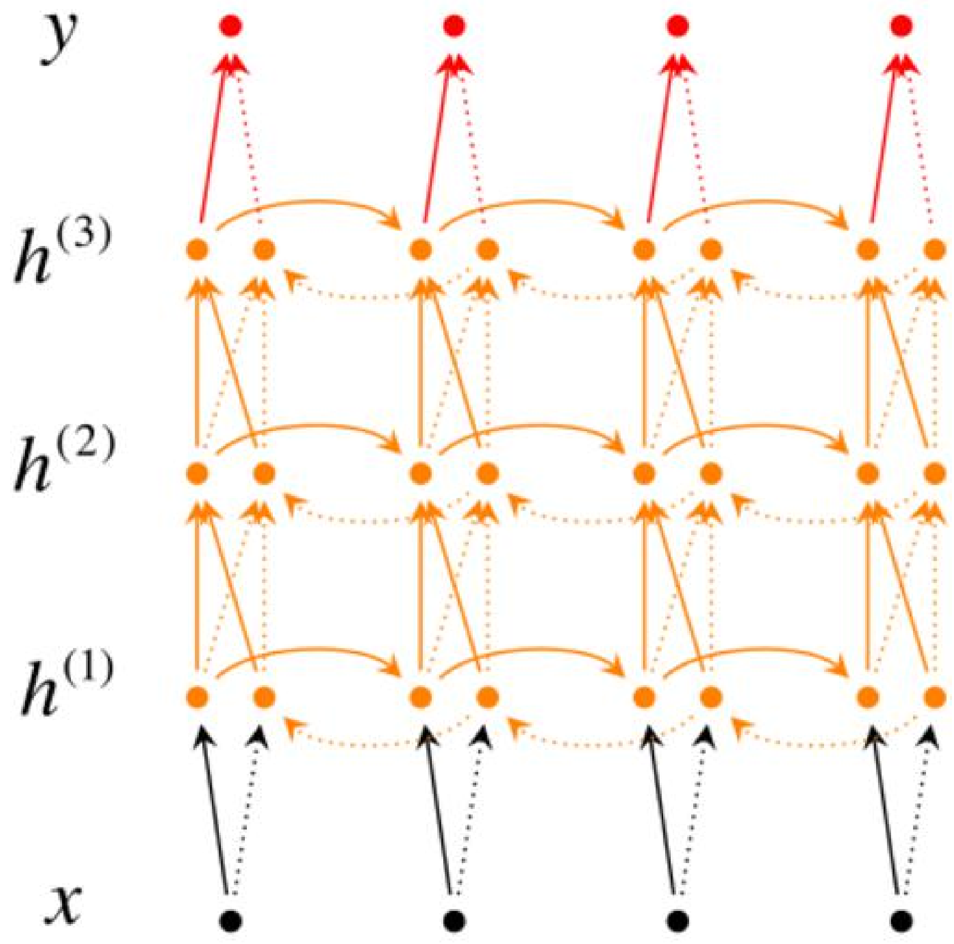


第l层网络的输入是第l-1层网络的输出。定义 $h_t^{(l)}$ 为在时刻t时第l层的隐状态:

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

其中 $U^{(l)}, W^{(l)}, b^{(l)}$ 为权重矩阵和偏置向量， $h_t^{(0)} = x_t$ 。

深层双向循环神经网络



我们把第i个隐藏层的值表示为 $h_t^{(i)}, h'^{(i)}_t$ ， 则**深度循环神经网络**的计算方式可以表示为：

$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}^{(i)} h_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)})$$

$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

$$\hat{y}_t = g(Uh_t + c) = g(U[\overrightarrow{h}_t^{(L)}; \overleftarrow{h}_t^{(L)}] + c)$$

主要参考文献

Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult[J]. IEEE transactions on neural networks, 1994, 5(2): 157-166.

Understanding LSTM Networks http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.

Jozefowicz R, Zaremba W, Sutskever I. An empirical exploration of recurrent network architectures[C]//International conference on machine learning. 2015: 2342-2350.

Greff K, Srivastava R K, Koutník J, et al. LSTM: A search space odyssey[J]. IEEE transactions on neural networks and learning systems, 2016, 28(10): 2222-2232.

LSTM Forward and Backward Pass http://arunmallya.github.io/writeups/nn/lstm/index.html#/

北京交通大学《深度学习》课程组成员

景丽萍：http://faculty.bjtu.edu.cn/8249/

桑基韬：http://faculty.bjtu.edu.cn/9129/

张淳杰：http://faculty.bjtu.edu.cn/9371/

万怀宇：http://faculty.bjtu.edu.cn/8793/

滕　竹：http://faculty.bjtu.edu.cn/8902/

原继东：http://faculty.bjtu.edu.cn/9076/

丛润民：http://faculty.bjtu.edu.cn/9374/

夏佳楠：http://faculty.bjtu.edu.cn/9430/

许万茹

杨　扩