



北京交通大学《深度学习》课件

第九讲 循环神经网络II



主讲教师：丛润民

北京交通大学 《深度学习》课程组





9.1 长程依赖问题

9.2 长短期记忆网络 (LSTM)

9.3 门控循环神经网络 (GRU)

9.4 深层循环神经网络



9.1 长程依赖问题

9.2 长短期记忆网络 (LSTM)

9.3 门控循环神经网络 (GRU)

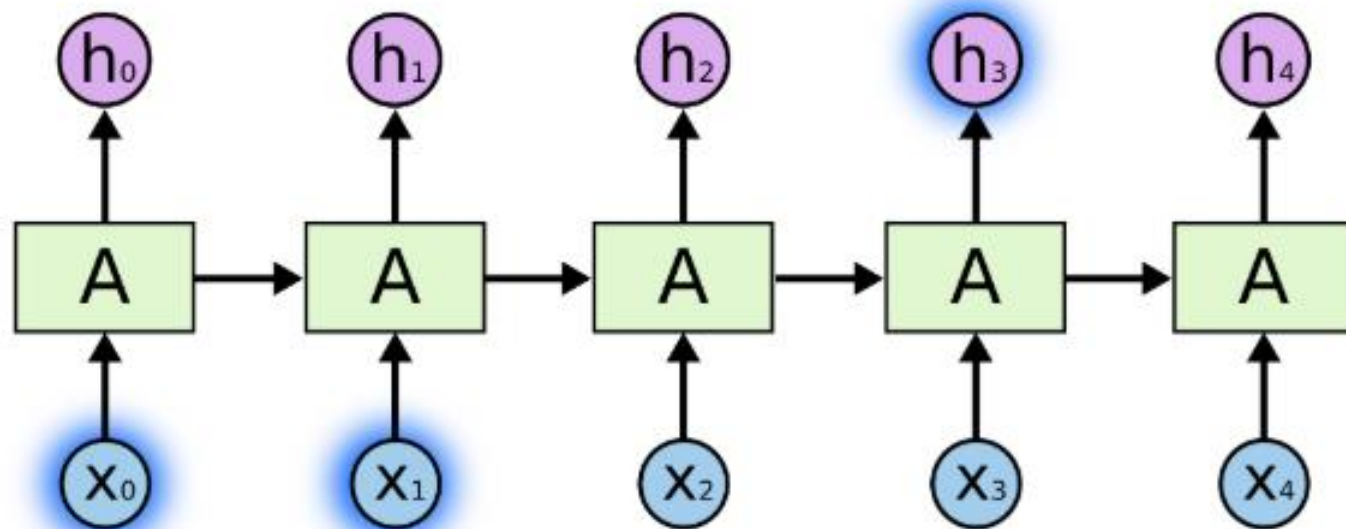
9.4 深层循环神经网络



9.1 长程依赖问题

RNN 的长处之一是它可以利用先前的信息到当前的任务上，尤其当相关的信息和预测的词之间的间隔较小时效果明显。

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



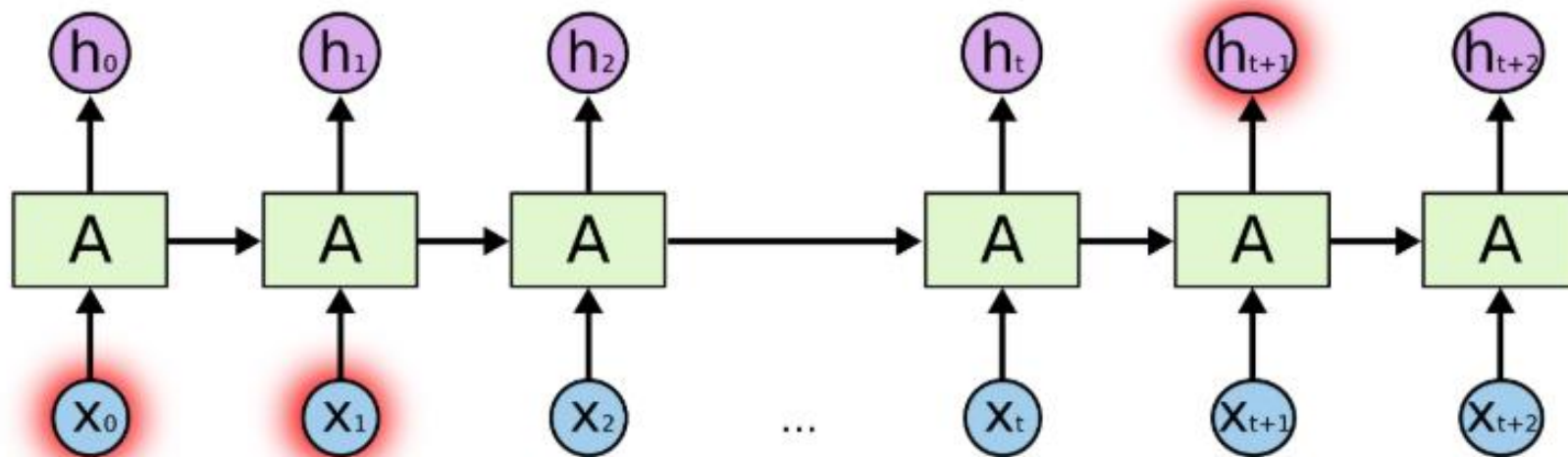
较近的相关信息或位置间隔



9.1 长程依赖问题

然而在间隔不断增大时，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



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？

上一节关于RNN的推导中，误差项沿时间反向传播的公式为：

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

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

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

其中， β_f 、 β_W 分别是对角矩阵和矩阵W模的上界。



9.1 长程依赖问题

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

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。

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，没法继续训练更新参数，也就是RNN的长程依赖问题。

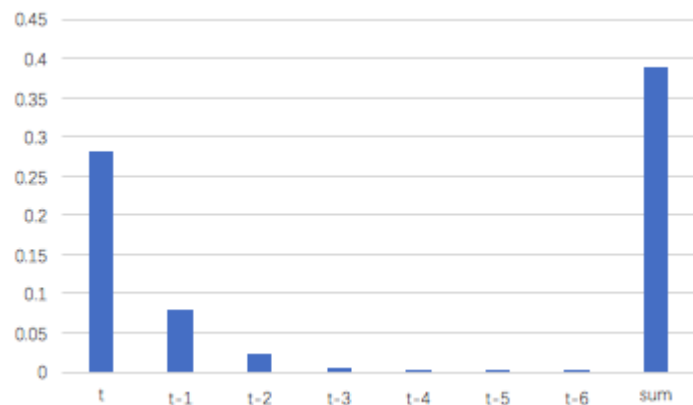


9.1 长程依赖问题

梯度消失举例：RNN中权重矩阵 W 最终的梯度是各个时刻的梯度之和，即：

$$\begin{aligned}\nabla_W E &= \sum_{k=1}^t \nabla_{W_k} E \\ &= \nabla_{W_t} E + \nabla_{W_{t-1}} E + \nabla_{W_{t-2}} E + \dots + \nabla_{W_1} E\end{aligned}$$

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。

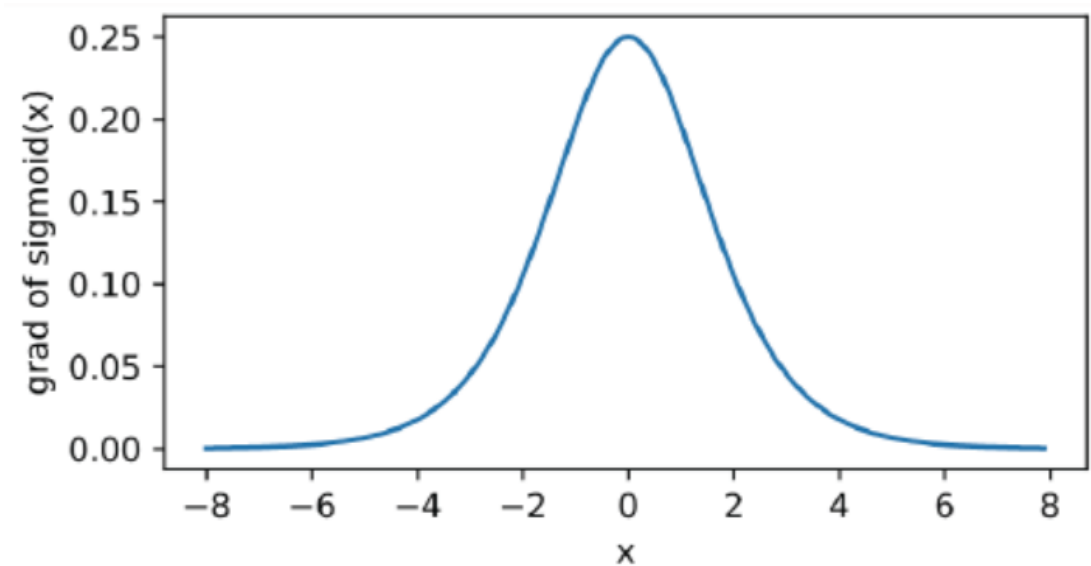
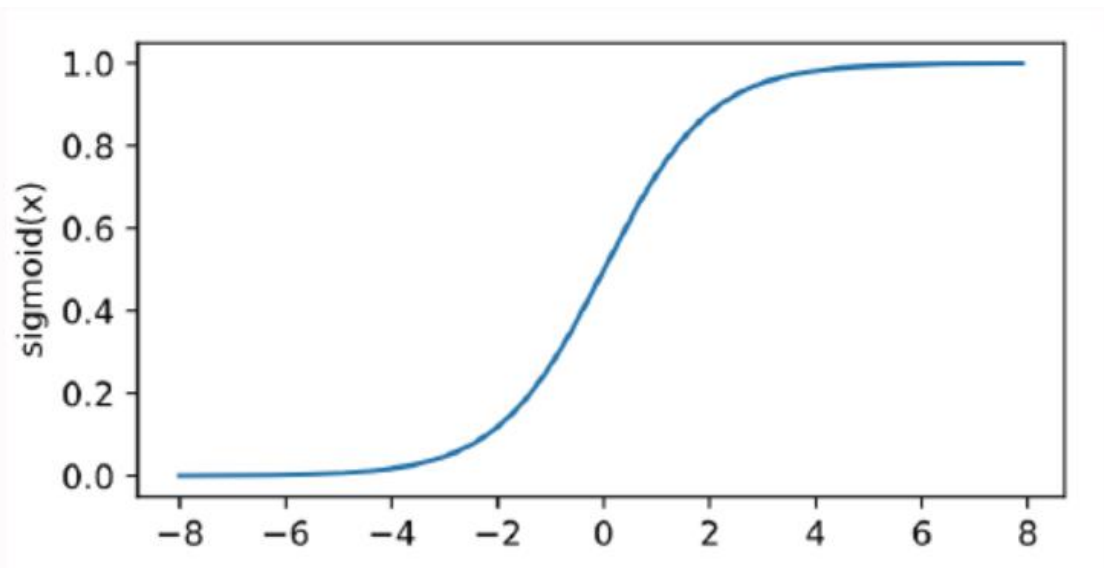
梯度消失更难检测，也更难处理一些。总的来说，有三种方法应对梯度消失问题：

1. 合理的初始化权重值。初始化权重，使每个神经元尽可能不要取极大或极小值，以躲开梯度消失的区域。
2. 使用relu代替sigmoid和tanh作为激活函数。



9.1 长程依赖问题

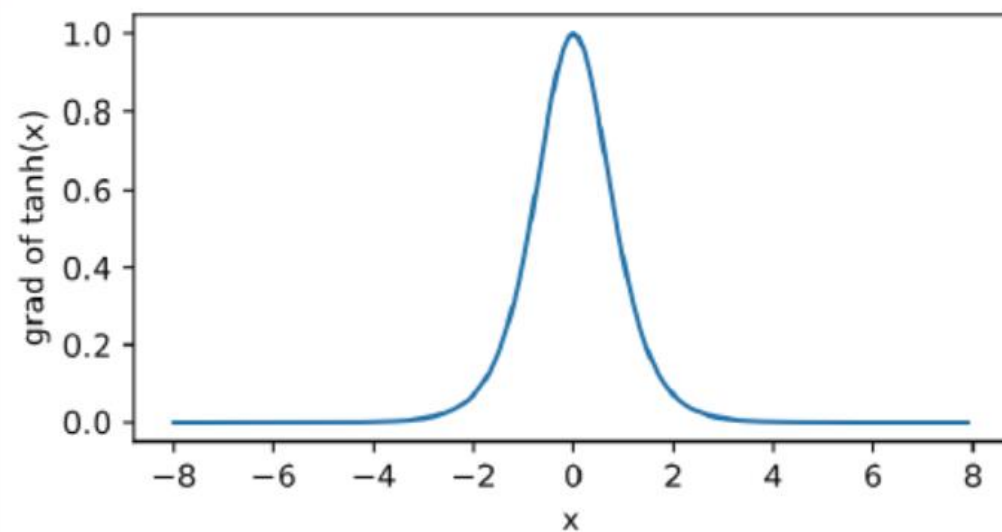
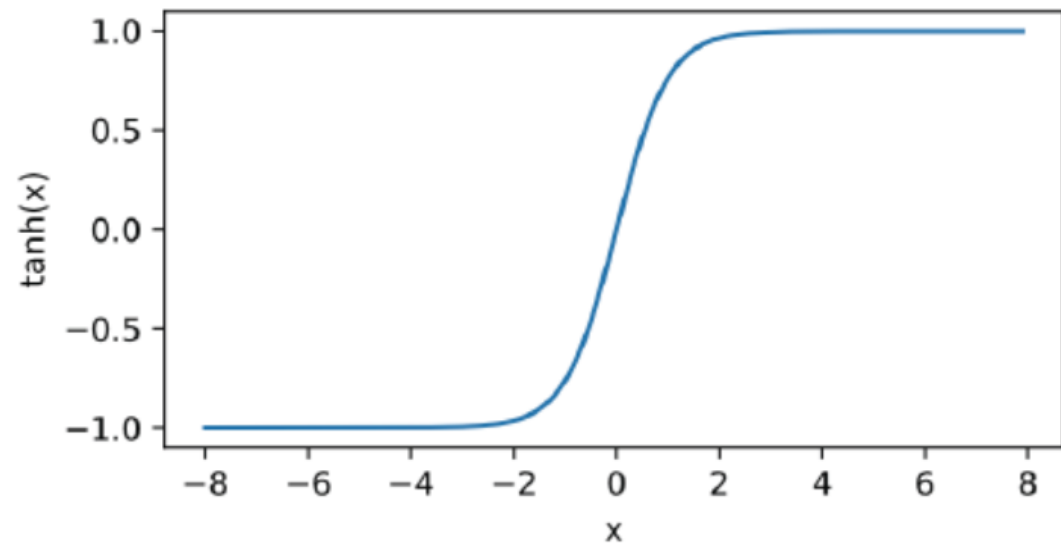
sigmoid函数的函数图和导数图





9.1 长程依赖问题

tanh函数的函数图和导数图





9.1 长程依赖问题

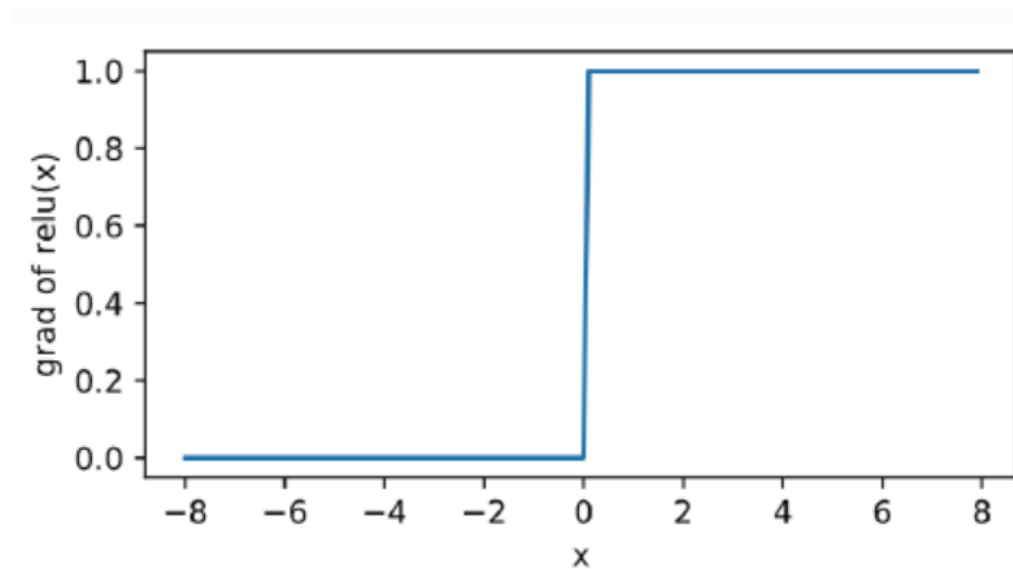
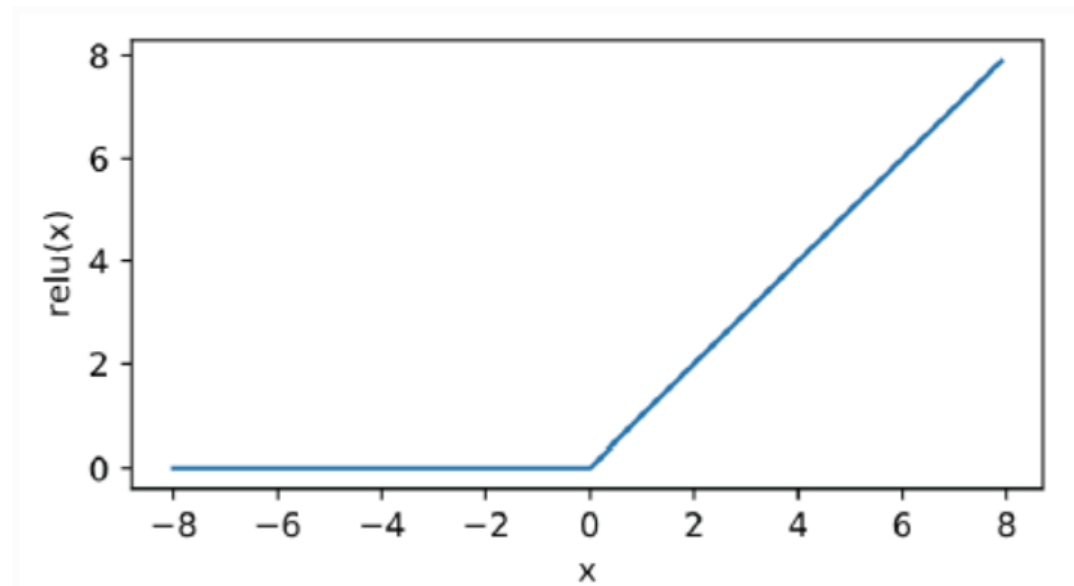
sigmoid函数与tanh函数比较：

- 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函数不是原点中心对称，输出均大于0
- tanh函数是原点中心对称，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。



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，也更难处理一些。总的来说，有三种方法应对梯度消失问题：

1. 合理的初始化权重值。初始化权重，使每个神经元尽可能不要取极大或极小值，以躲开梯度消失的区域。
2. 使用relu代替sigmoid和tanh作为激活函数。
3. 使用其他结构的RNNs，比如长短时记忆网络（LSTM）和 Gated Recurrent Unit（GRU）。

接下来将重点介绍LSTM和GRU两种网络。



9.1 长程依赖问题

9.2 长短期记忆网络 (LSTM)

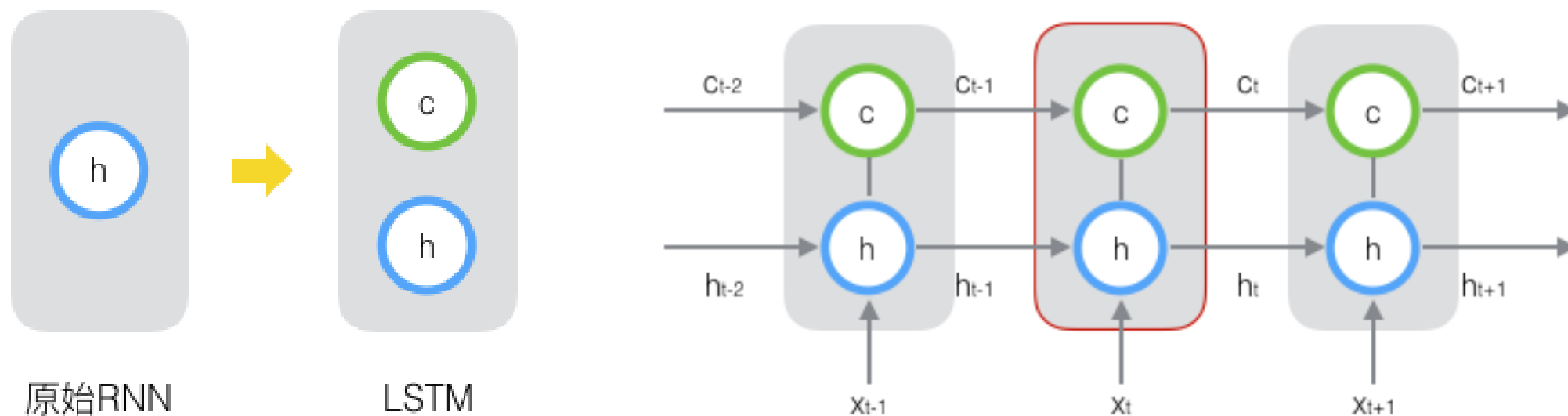
9.3 门控循环神经网络 (GRU)

9.4 深层循环神经网络



9.2 长短期记忆网络 (LSTM)

Long Short Term Memory networks (以下简称LSTMs)，一种特殊的RNN网络，该网络设计出来是为了解决长程依赖问题。



增加状态 c ，称为单元状态(cell state)，让它来保存长期的状态



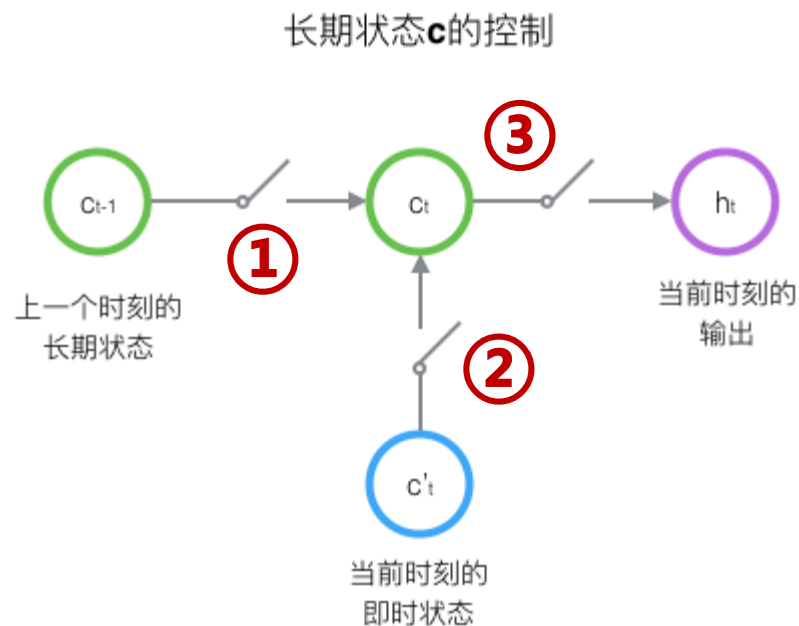
9.2 长短期记忆网络 (LSTM)

LSTM的关键，就是怎样控制长期状态 c 。LSTM使用三个控制开关：

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 ；

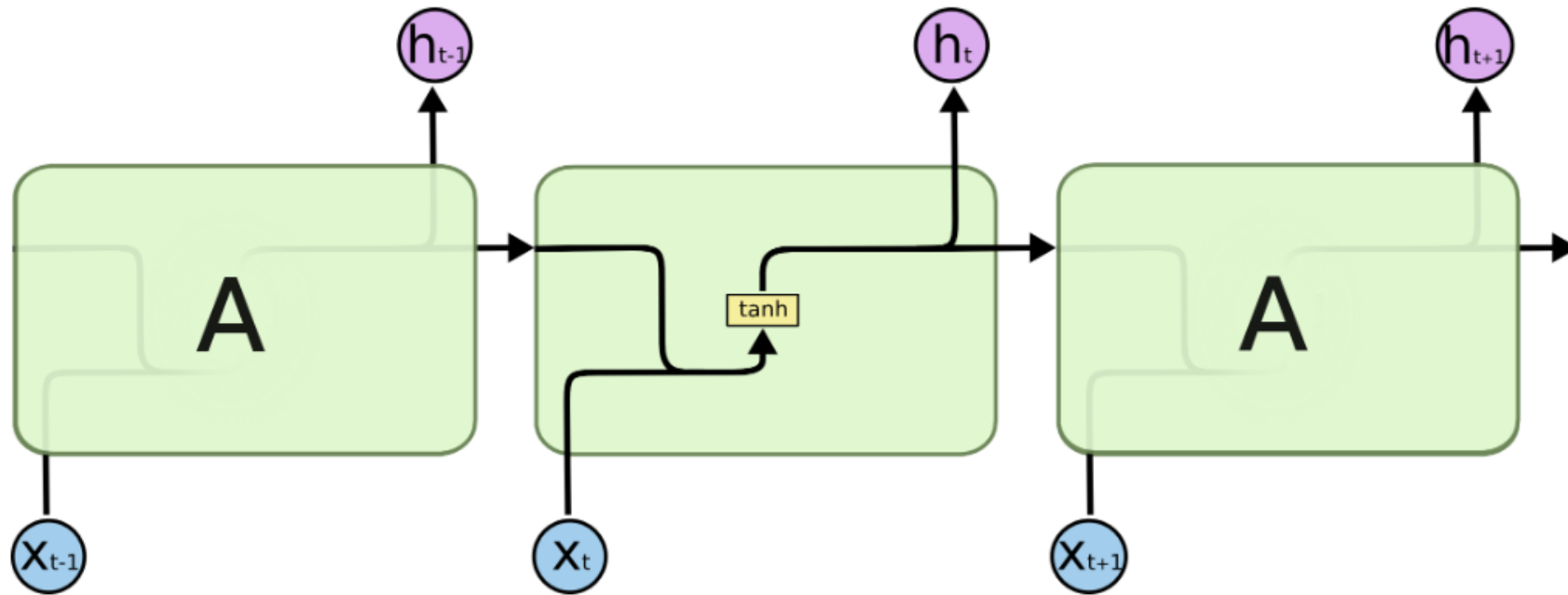
第三个开关，负责控制是否把长期状态 c 作为当前的LSTM的输出；





9.2 长短期记忆网络 (LSTM)

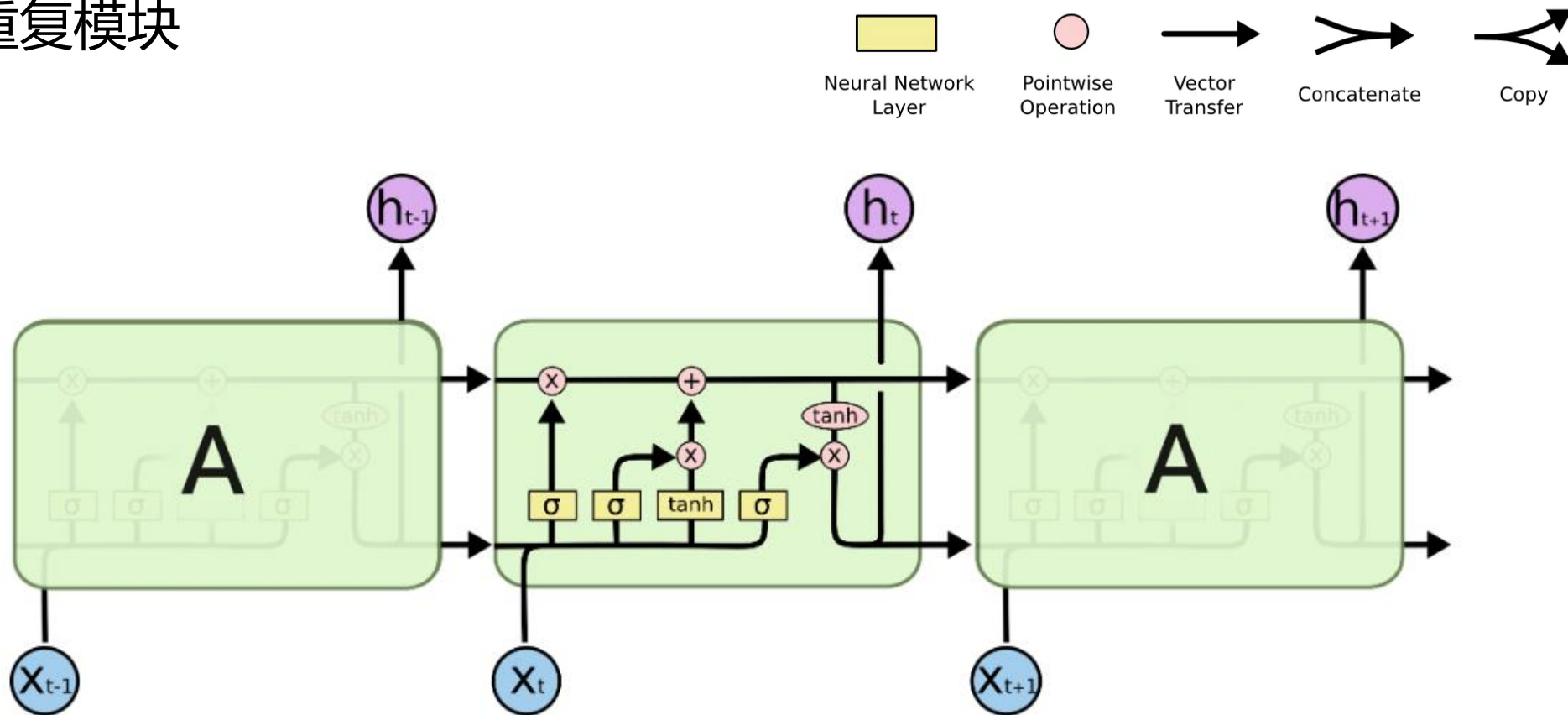
标准RNN的重复模块





9.2 长短期记忆网络 (LSTM)

LSTM 的重复模块



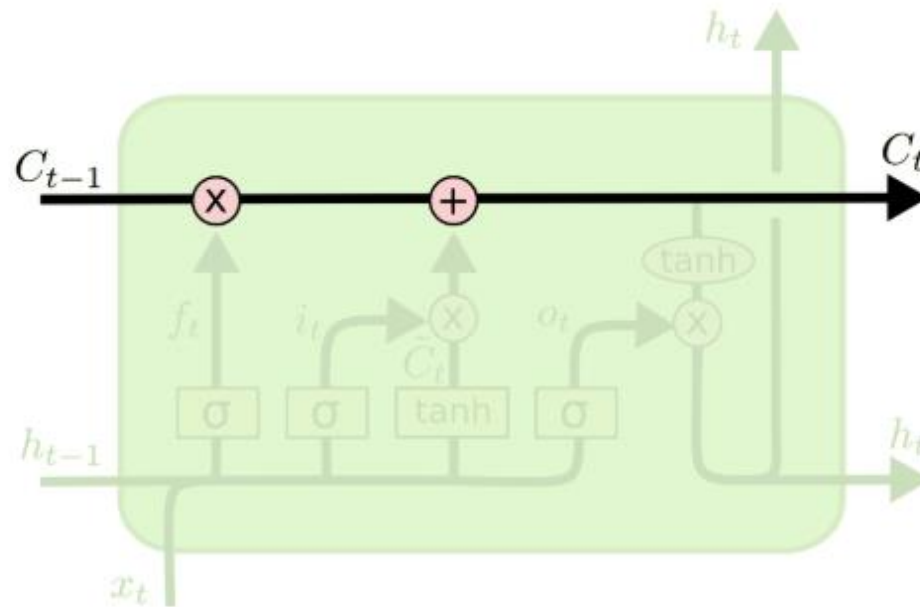
除了 h 在随时间流动，单元状态 c 也在随时间流动，单元状态 c 就代表着长期记忆。



9.2 长短期记忆网络 (LSTM)

LSTM 的核心思想

LSTM 的关键是单元状态，如水平线在图上方贯穿运行。



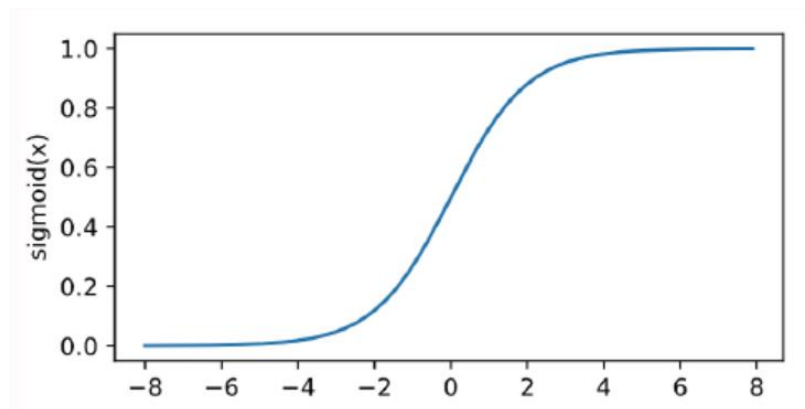
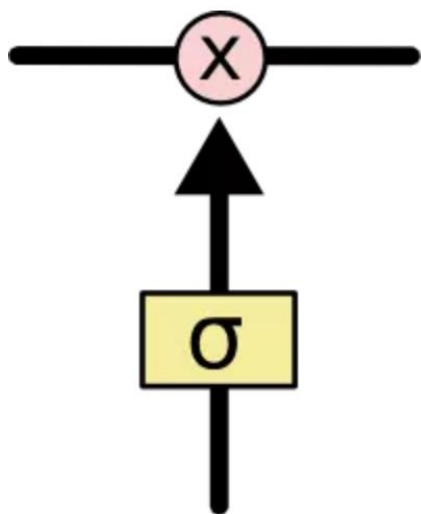
单元状态的传递类似于传送带，其直接在整个链上运行，中间只有一些少量的线性交互，容易保存相关信息。



9.2 长短期记忆网络 (LSTM)

前面描述的开关是怎样在算法中实现的呢？

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



9.2 长短期记忆网络 (LSTM)

- LSTM用两个门来控制单元状态 c 的内容
 - **遗忘门 (forget gate)** , 它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 ;
 - **输入门 (input gate)** , 它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



9.2 长短期记忆网络 (LSTM)

逐步理解 LSTM之遗忘门

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

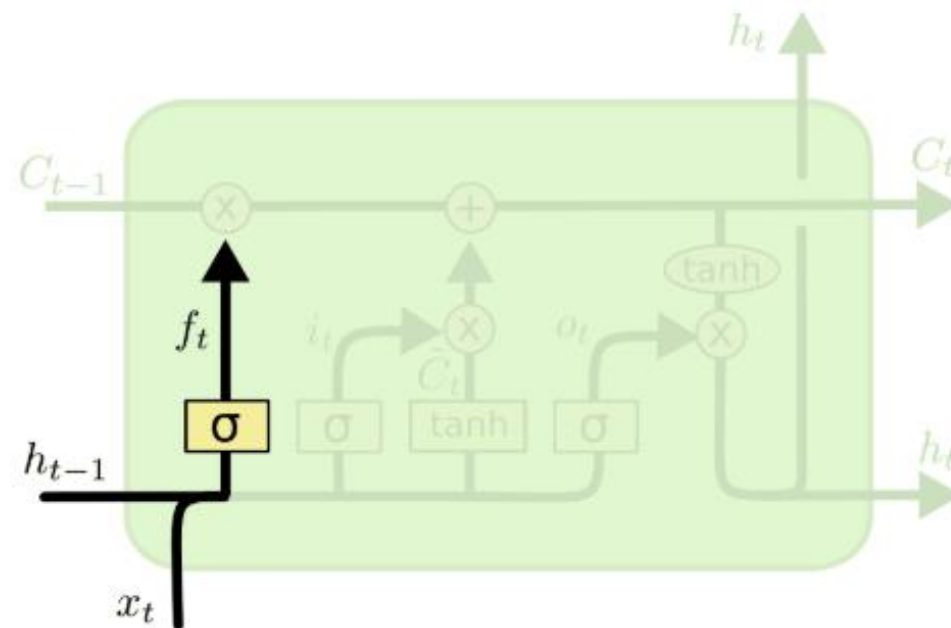
上式中, W_f 是遗忘门的权重矩阵

$[h_{t-1}, x_t]$ 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 $d_c \times (d_h + d_x)$





9.2 长短期记忆网络 (LSTM)

逐步理解 LSTM之遗忘门

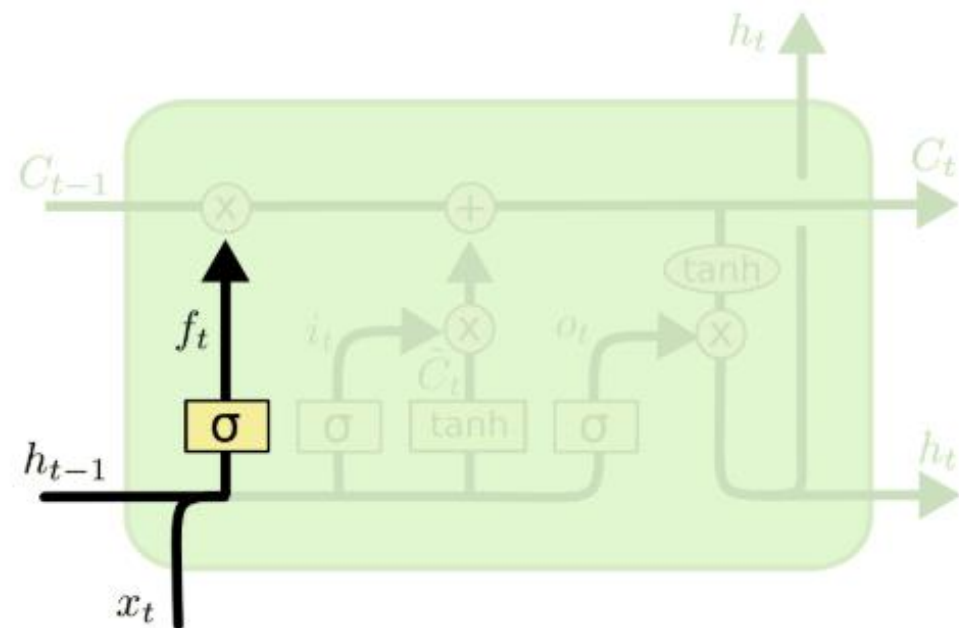
$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

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

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 $d_c \times d_x$ 。

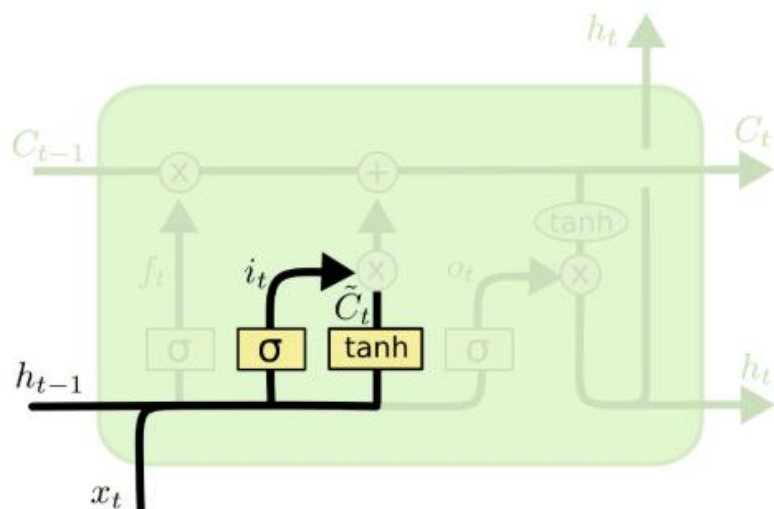


这个门怎么做到“遗忘”的呢？怎么理解？既然是遗忘旧的内容，为什么这个门还要接收新的 x_t ？



9.2 长短期记忆网络 (LSTM)

逐步理解 LSTM之输入门



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

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (\text{式3})$$

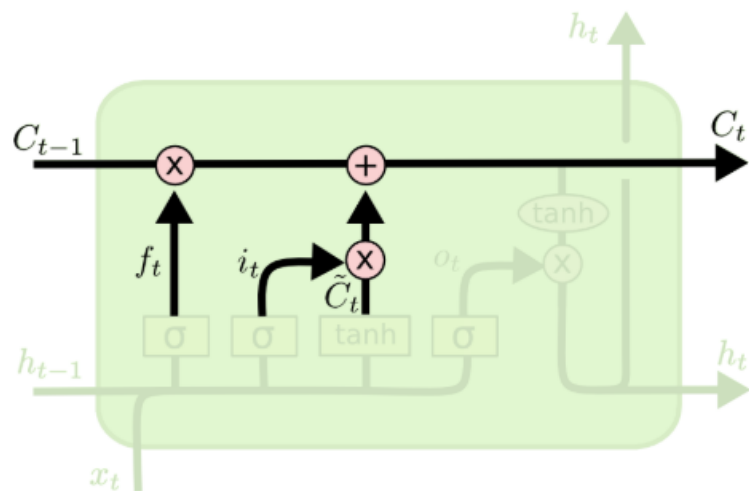
sigmoid 函数称为输入门，决定将要更新什么值

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



9.2 长短期记忆网络 (LSTM)

逐步理解 LSTM之更新单元状态



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (\text{式4})$$

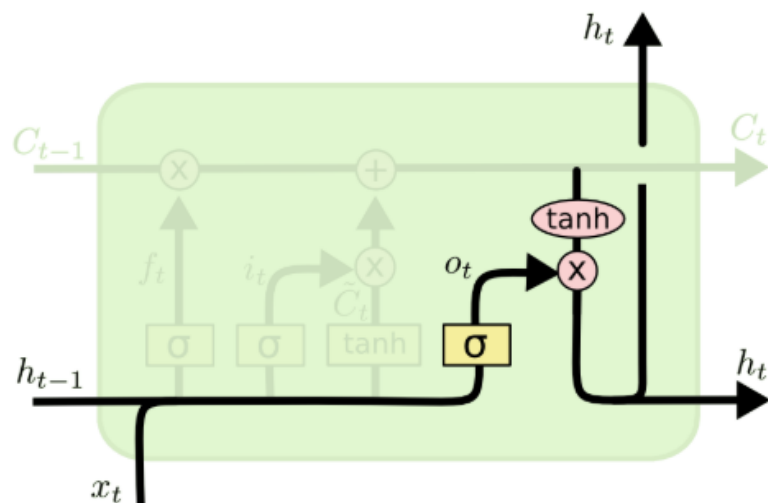
现在开始计算当前时刻的单元状态 C_t 。它是由上一次的单元状态 C_{t-1} 按元素乘以遗忘门 f_t ，再用当前输入的单元状态 \tilde{C}_t 按元素乘以输入门 i_t ，再将两个积加和产生的。

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。



9.2 长短期记忆网络 (LSTM)

逐步理解 LSTM之**输出门**



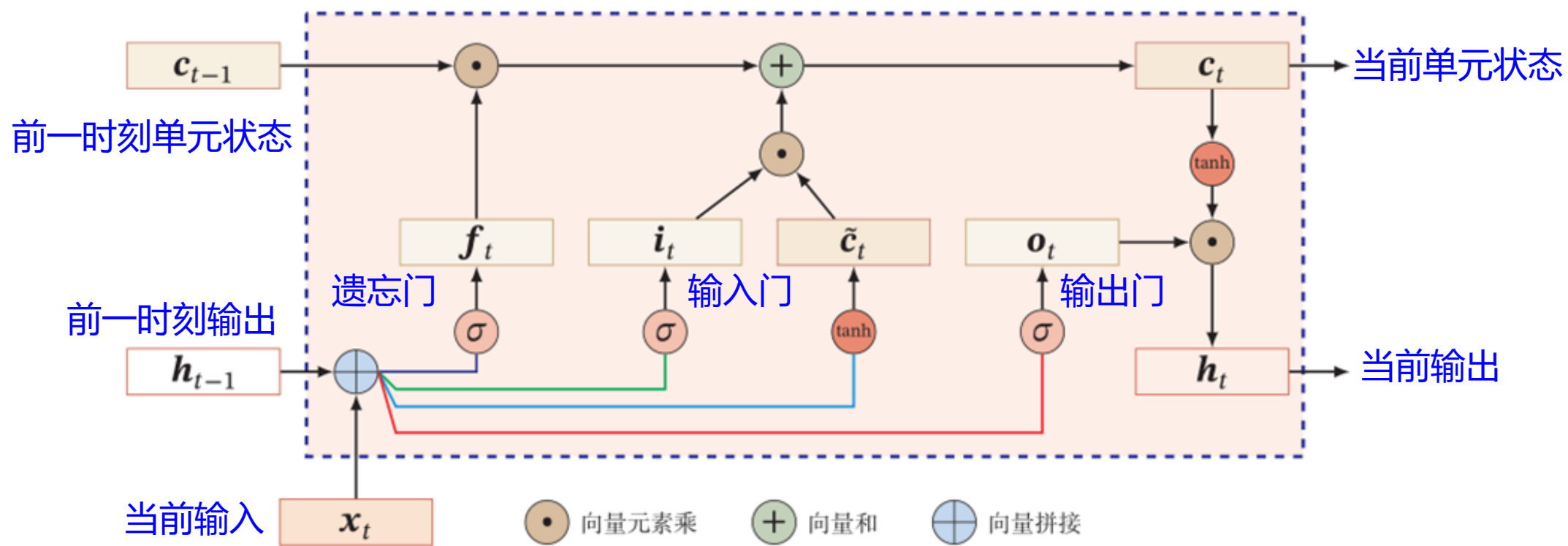
$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o) \quad (\text{式5})$$

$$h_t = o_t * \tanh(C_t) \quad (\text{式6})$$

输出门控制了长期记忆对当前输出的影响，其由输出门和单元状态共同确定。



9.2 长短期记忆网络 (LSTM)



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

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c)$$

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

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

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

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



9.2 长短期记忆网络 (LSTM)

1. 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：

- ① 前向计算每个神经元的输出值，对于LSTM来说，即 $\mathbf{f}_t, \mathbf{i}_t, \mathbf{c}_t, \mathbf{o}_t, \mathbf{h}_t$ 五个向量的值。计算方法已经在上一节中描述过了。
- ② 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。
- ③ 根据相应的误差项，计算每个权重的梯度。



9.2 长短期记忆网络 (LSTM)

2. 关于公式和符号的说明

设定门gate的激活函数为sigmoid函数，输出的激活函数为tanh函数。它们的导数分别为：

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

$$\tanh(z) = y = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$
$$\tanh'(z) = 1 - y^2$$

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



9.2 长短期记忆网络 (LSTM)

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 、
 计算单元状态的权重矩阵 W_c 和偏置项 b_c 。

按元素乘 \circ 符号。当 \circ 作用于两个**向量**时：

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

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

$$W_{fh} \ W_{fx} \ W_{ih} \ W_{ix} \ W_{oh} \ W_{ox} \ W_{ch} \ W_{cx}$$

当 \circ 作用于一个**向量**和一个**矩阵**时：

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



9.2 长短期记忆网络 (LSTM)

当 \circ 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：

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

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：

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

上面这两点，在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 。定义t时刻的误差项 δ_t 为：
$$\delta_t \stackrel{\text{def}}{=} \frac{\partial E}{\partial \mathbf{h}_t}$$

LSTM有四个加权输入，分别对应 $\mathbf{f}_t, \mathbf{i}_t, \mathbf{c}_t, \mathbf{o}_t$ ，希望往上一层传递一个误差项而不是四个。但仍然需要定义出这四个加权输入，以及他们对应的误差项。

$$\begin{aligned}\mathbf{net}_{f,t} &= W_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f \\ &= W_{fh}\mathbf{h}_{t-1} + W_{fx}\mathbf{x}_t + \mathbf{b}_f\end{aligned}$$

$$\begin{aligned}\mathbf{net}_{i,t} &= W_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i \\ &= W_{ih}\mathbf{h}_{t-1} + W_{ix}\mathbf{x}_t + \mathbf{b}_i\end{aligned}$$

$$\begin{aligned}\mathbf{net}_{\tilde{c},t} &= W_c[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c \\ &= W_{ch}\mathbf{h}_{t-1} + W_{cx}\mathbf{x}_t + \mathbf{b}_c\end{aligned}$$

$$\begin{aligned}\mathbf{net}_{o,t} &= W_o[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o \\ &= W_{oh}\mathbf{h}_{t-1} + W_{ox}\mathbf{x}_t + \mathbf{b}_o\end{aligned}$$

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

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

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

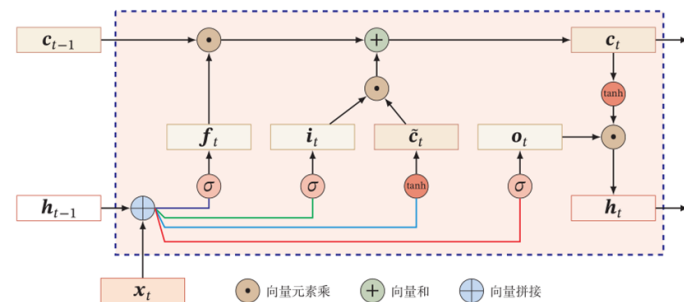
$$\delta_{o,t} \stackrel{\text{def}}{=} \frac{\partial E}{\partial \mathbf{net}_{o,t}}$$



9.2 长短期记忆网络 (LSTM)

3. 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}



$$\delta_{t-1}^T = \frac{\partial E}{\partial \mathbf{h}_{t-1}} = \frac{\partial E}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}} = \delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}}$$

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, 我们列出 \mathbf{h}_t 的计算公式, 即前面的**式6**和**式4**:

$$\mathbf{h}_t = \mathbf{o}_t \circ \tanh(\mathbf{c}_t)$$

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

利用全导数公式可得 (式7) :

$$\delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}} = \delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{o}_t} \frac{\partial \mathbf{o}_t}{\partial \text{net}_{o,t}} \frac{\partial \text{net}_{o,t}}{\partial \mathbf{h}_{t-1}} + \delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{c}_t} \frac{\partial \mathbf{c}_t}{\partial \mathbf{f}_t} \frac{\partial \mathbf{f}_t}{\partial \text{net}_{f,t}} \frac{\partial \text{net}_{f,t}}{\partial \mathbf{h}_{t-1}} + \delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{c}_t} \frac{\partial \mathbf{c}_t}{\partial \mathbf{i}_t} \frac{\partial \mathbf{i}_t}{\partial \text{net}_{i,t}} \frac{\partial \text{net}_{i,t}}{\partial \mathbf{h}_{t-1}}$$



9.2 长短期记忆网络 (LSTM)

$$\delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}} = \delta_t^T \overset{\text{😊}}{\frac{\partial \mathbf{h}_t}{\partial \mathbf{o}_t}} \frac{\partial \mathbf{o}_t}{\partial \text{net}_{o,t}} \frac{\partial \text{net}_{o,t}}{\partial \mathbf{h}_{t-1}} + \delta_t^T \overset{\text{😊}}{\frac{\partial \mathbf{h}_t}{\partial \mathbf{c}_t}} \overset{\text{😊}}{\frac{\partial \mathbf{c}_t}{\partial \mathbf{f}_t}} \frac{\partial \mathbf{f}_t}{\partial \text{net}_{f,t}} \frac{\partial \text{net}_{f,t}}{\partial \mathbf{h}_{t-1}} + \delta_t^T \overset{\text{😊}}{\frac{\partial \mathbf{h}_t}{\partial \mathbf{c}_t}} \overset{\text{😊}}{\frac{\partial \mathbf{c}_t}{\partial \mathbf{i}_t}} \frac{\partial \mathbf{i}_t}{\partial \text{net}_{i,t}} \frac{\partial \text{net}_{i,t}}{\partial \mathbf{h}_{t-1}}$$

下面，要把**式7**中的每个偏导数都求出来。

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

根据**式6**，可以求出：

$$\begin{aligned}\frac{\partial \mathbf{h}_t}{\partial \mathbf{o}_t} &= \text{diag}[\tanh(\mathbf{c}_t)] \\ \frac{\partial \mathbf{h}_t}{\partial \mathbf{c}_t} &= \text{diag}\left[\mathbf{o}_t \circ \left(1 - \tanh(\mathbf{c}_t)^2\right)\right]\end{aligned}$$

根据**式4**，可以求出：

$$\begin{aligned}\frac{\partial \mathbf{c}_t}{\partial \mathbf{f}_t} &= \text{diag}[\mathbf{c}_{t-1}] \\ \frac{\partial \mathbf{c}_t}{\partial \mathbf{i}_t} &= \text{diag}[\tilde{\mathbf{c}}_t] \\ \frac{\partial \mathbf{c}_t}{\partial \tilde{\mathbf{c}}_t} &= \text{diag}[\mathbf{i}_t]\end{aligned}$$



9.2 长短期记忆网络 (LSTM)

因为:

$$\mathbf{o}_t = \sigma(\mathbf{net}_{o,t})$$

$$\mathbf{net}_{o,t} = W_{oh}\mathbf{h}_{t-1} + W_{ox}\mathbf{x}_t + \mathbf{b}_o$$

$$\mathbf{f}_t = \sigma(\mathbf{net}_{f,t})$$

$$\mathbf{net}_{f,t} = W_{fh}\mathbf{h}_{t-1} + W_{fx}\mathbf{x}_t + \mathbf{b}_f$$

$$\mathbf{i}_t = \sigma(\mathbf{net}_{i,t})$$

$$\mathbf{net}_{i,t} = W_{ih}\mathbf{h}_{t-1} + W_{ix}\mathbf{x}_t + \mathbf{b}_i$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{net}_{\tilde{c},t})$$

$$\mathbf{net}_{\tilde{c},t} = W_{ch}\mathbf{h}_{t-1} + W_{cx}\mathbf{x}_t + \mathbf{b}_c$$

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



$$\frac{\partial \mathbf{o}_t}{\partial \mathbf{net}_{o,t}} = \text{diag}[\mathbf{o}_t \circ (1 - \mathbf{o}_t)]$$

$$\frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h}_{t-1}} = W_{oh}$$

$$\frac{\partial \mathbf{f}_t}{\partial \mathbf{net}_{f,t}} = \text{diag}[\mathbf{f}_t \circ (1 - \mathbf{f}_t)]$$

$$\frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h}_{t-1}} = W_{fh}$$

$$\frac{\partial \mathbf{i}_t}{\partial \mathbf{net}_{i,t}} = \text{diag}[\mathbf{i}_t \circ (1 - \mathbf{i}_t)]$$

$$\frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h}_{t-1}} = W_{ih}$$

$$\frac{\partial \tilde{\mathbf{c}}_t}{\partial \mathbf{net}_{\tilde{c},t}} = \text{diag}[1 - \tilde{\mathbf{c}}_t^2]$$

$$\frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{h}_{t-1}} = W_{ch}$$



9.2 长短期记忆网络 (LSTM)

根据 $\delta_{o,t}$ $\delta_{f,t}$ $\delta_{i,t}$ $\delta_{\tilde{c},t}$ 的定义, 可知: $\delta_{o,t} \stackrel{\text{def}}{=} \frac{\partial E}{\partial \text{net}_{o,t}} = \frac{\partial E}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{o}_t} \frac{\partial \mathbf{o}_t}{\partial \text{net}_{o,t}}$

$$\frac{\partial \mathbf{h}_t}{\partial \mathbf{o}_t} = \text{diag}[\tanh(\mathbf{c}_t)] \quad \frac{\partial \mathbf{o}_t}{\partial \text{net}_{o,t}} = \text{diag}[\mathbf{o}_t \circ (1 - \mathbf{o}_t)]$$

$$\delta_{o,t}^T = \delta_t^T \circ \tanh(\mathbf{c}_t) \circ \mathbf{o}_t \circ (1 - \mathbf{o}_t) \quad (\text{式8})$$

$$\delta_{f,t}^T = \delta_t^T \circ \mathbf{o}_t \circ \left(1 - \tanh(\mathbf{c}_t)^2\right) \circ \mathbf{c}_{t-1} \circ \mathbf{f}_t \circ (1 - \mathbf{f}_t) \quad (\text{式9})$$

$$\delta_{i,t}^T = \delta_t^T \circ \mathbf{o}_t \circ \left(1 - \tanh(\mathbf{c}_t)^2\right) \circ \tilde{\mathbf{c}}_t \circ \mathbf{i}_t \circ (1 - \mathbf{i}_t) \quad (\text{式10})$$

$$\delta_{\tilde{c},t}^T = \delta_t^T \circ \mathbf{o}_t \circ \left(1 - \tanh(\mathbf{c}_t)^2\right) \circ \mathbf{i}_t \circ (1 - \tilde{\mathbf{c}}^2) \quad (\text{式11})$$



9.2 长短期记忆网络 (LSTM)

将上述偏导数带入到**式7**，得到：

$$\begin{aligned}\delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}} &= \delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{o}_t} \frac{\partial \mathbf{o}_t}{\partial \mathbf{net}_{o,t}} \frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h}_{t-1}} + \delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{c}_t} \frac{\partial \mathbf{c}_t}{\partial \mathbf{f}_t} \frac{\partial \mathbf{f}_t}{\partial \mathbf{net}_{f,t}} \frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h}_{t-1}} + \delta_t^T \frac{\partial \mathbf{h}_t}{\partial \mathbf{c}_t} \frac{\partial \mathbf{c}_t}{\partial \mathbf{i}_t} \frac{\partial \mathbf{i}_t}{\partial \mathbf{net}_{i,t}} \frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h}_{t-1}} \\&= \delta_{o,t}^T \frac{\partial \mathbf{net}_{o,t}}{\partial \mathbf{h}_{t-1}} + \delta_{f,t}^T \frac{\partial \mathbf{net}_{f,t}}{\partial \mathbf{h}_{t-1}} + \delta_{i,t}^T \frac{\partial \mathbf{net}_{i,t}}{\partial \mathbf{h}_{t-1}} + \delta_{\tilde{c},t}^T \frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial \mathbf{h}_{t-1}} \\&= \delta_{o,t}^T W_{oh} + \delta_{f,t}^T W_{fh} + \delta_{i,t}^T W_{ih} + \delta_{\tilde{c},t}^T W_{ch} = \delta_{t-1}^T\end{aligned}\tag{式12}$$



9.2 长短期记忆网络 (LSTM)

式12就是将误差沿时间反向传播一个时刻的公式。有了它，可以写出将误差项向前传递到任意 k 时刻的公式：

$$\delta_k^T = \prod_{j=k}^{t-1} \delta_{o,j}^T W_{oh} + \delta_{f,j}^T W_{fh} + \delta_{i,j}^T W_{ih} + \delta_{\tilde{c},j}^T W_{ch} \quad (\text{式13})$$



9.2 长短期记忆网络 (LSTM)

4. 将误差项传递到上一层

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:

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

本次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 $\text{net}_{f,t}^l, \text{net}_{i,t}^l, \text{net}_{\tilde{c},t}^l, \text{net}_{o,t}^l$ 都是 x_t 的函数, 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:

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

式14就是将误差传递到上一层的公式。

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



9.2 长短期记忆网络 (LSTM)

5. 权重梯度的计算

对于 $W_{fh}, W_{ih}, W_{ch}, W_{oh}$ 的权重梯度，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。

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

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

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

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

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

$$\begin{aligned} \frac{\partial E}{\partial W_{oh}} &= \sum_{j=1}^t \delta_{o,j} \mathbf{h}_{j-1}^T \\ \frac{\partial E}{\partial W_{fh}} &= \sum_{j=1}^t \delta_{f,j} \mathbf{h}_{j-1}^T \\ \frac{\partial E}{\partial W_{ih}} &= \sum_{j=1}^t \delta_{i,j} \mathbf{h}_{j-1}^T \\ \frac{\partial E}{\partial W_{ch}} &= \sum_{j=1}^t \delta_{\tilde{c},j} \mathbf{h}_{j-1}^T \end{aligned}$$



9.2 长短期记忆网络 (LSTM)

对于偏置项 $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_c, \mathbf{b}_o$ 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：

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

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

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



9.2 长短期记忆网络 (LSTM)

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

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

$$\begin{aligned}
 \frac{\partial E}{\partial W_{ox}} &= \frac{\partial E}{\partial \mathbf{net}_{o,t}} \frac{\partial \mathbf{net}_{o,t}}{\partial W_{ox}} \\
 &= \delta_{o,t} \mathbf{x}_t^T \\
 \frac{\partial E}{\partial W_{fx}} &= \frac{\partial E}{\partial \mathbf{net}_{f,t}} \frac{\partial \mathbf{net}_{f,t}}{\partial W_{fx}} \\
 &= \delta_{f,t} \mathbf{x}_t^T \\
 \frac{\partial E}{\partial W_{ix}} &= \frac{\partial E}{\partial \mathbf{net}_{i,t}} \frac{\partial \mathbf{net}_{i,t}}{\partial W_{ix}} \\
 &= \delta_{i,t} \mathbf{x}_t^T \\
 \frac{\partial E}{\partial W_{cx}} &= \frac{\partial E}{\partial \mathbf{net}_{\tilde{c},t}} \frac{\partial \mathbf{net}_{\tilde{c},t}}{\partial W_{cx}} \\
 &= \delta_{\tilde{c},t} \mathbf{x}_t^T
 \end{aligned}$$



9.1 长程依赖问题

9.2 长短期记忆网络 (LSTM)

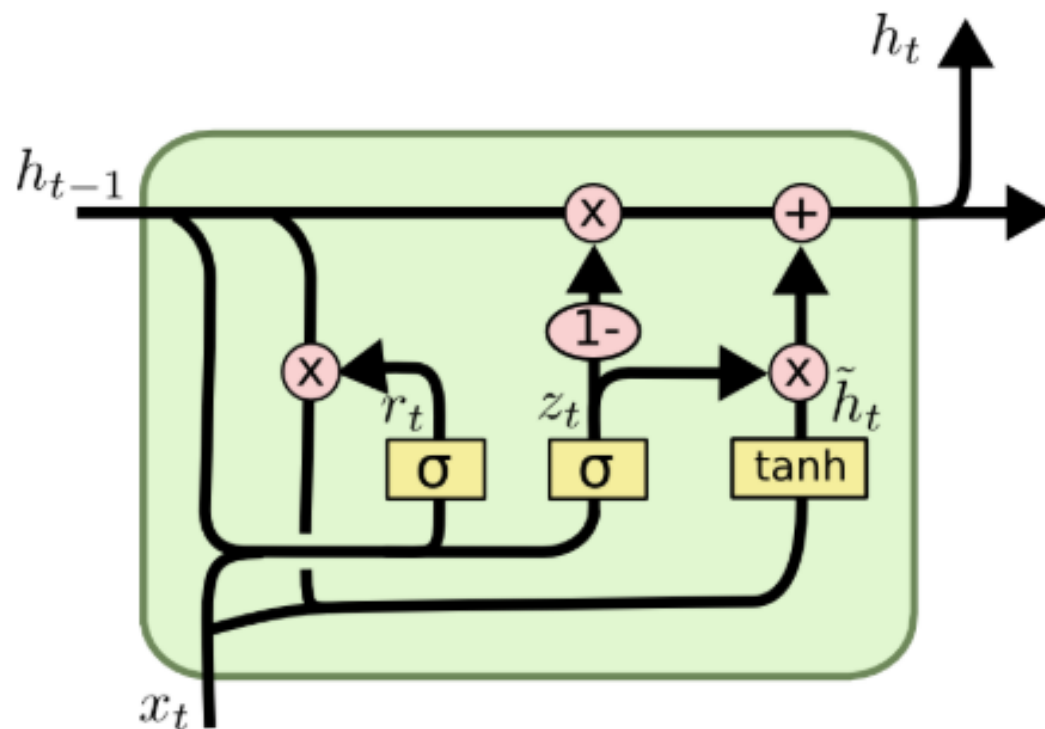
9.3 门控循环神经网络 (GRU)

9.4 深层循环神经网络



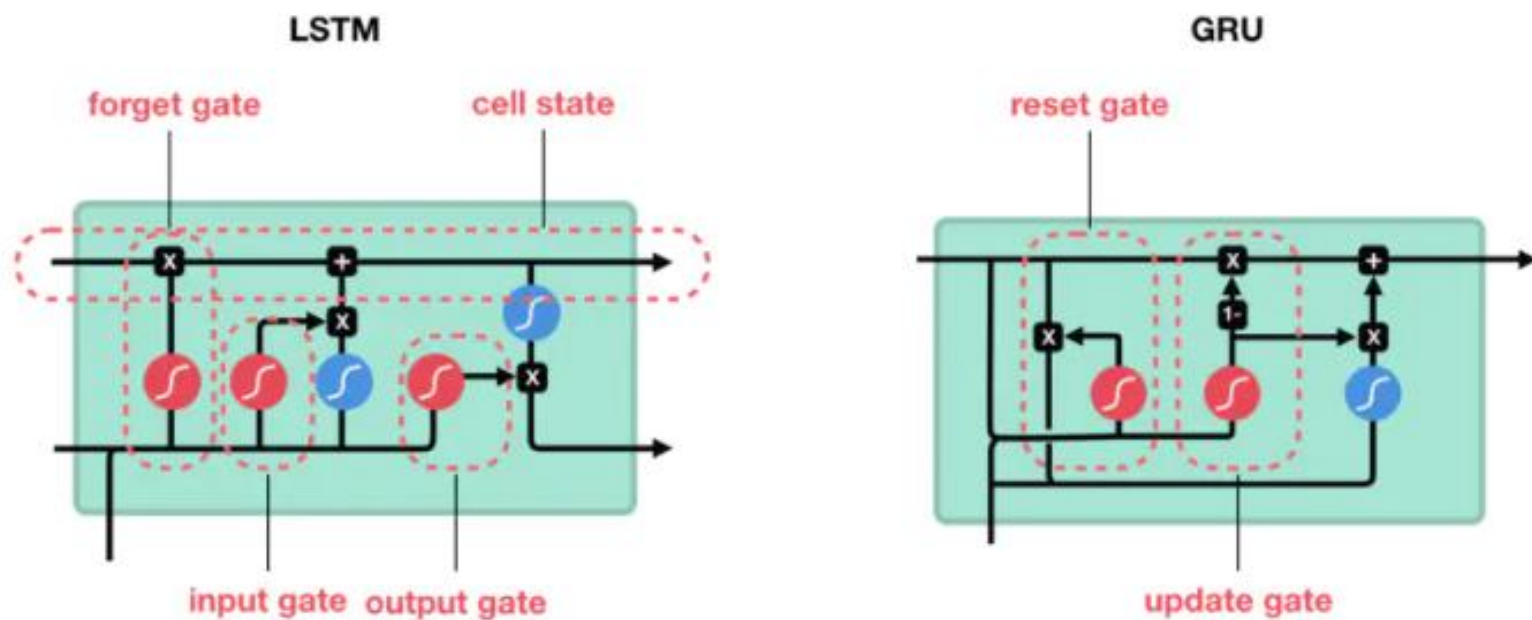
9.3 门控循环神经网络 (GRU)

GRU (Gate Recurrent Unit) 是循环神经网络RNN的一种。和LSTM一样，也是为了解决长期记忆和反向传播中的梯度等问题而提出来的。





9.3 门控循环神经网络 (GRU)



GRU是LSTM的一种变体，它较LSTM网络的结构更加简单，而且效果也很好。

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。而在GRU模型中只有两个门，分别是更新门和重置门。

另外，GRU将单元状态与输出合并为一个状态 h 。

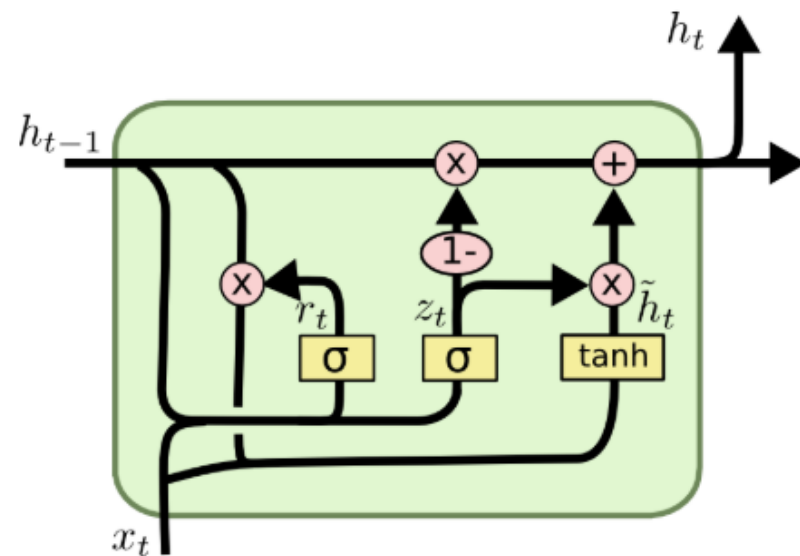


9.3 门控循环神经网络 (GRU)

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，更新门的值越大说明前一时刻的状态信息带入越多。

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



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

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

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

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



9.3 门控循环神经网络 (GRU)

LSTM与GRU

- GRU的参数更少，因而训练稍快或需要更少的数据来泛化。
- 如果你有足够的数据，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.



9.1 长程依赖问题

9.2 长短期记忆网络 (LSTM)

9.3 门控循环神经网络 (GRU)

9.4 深层循环神经网络



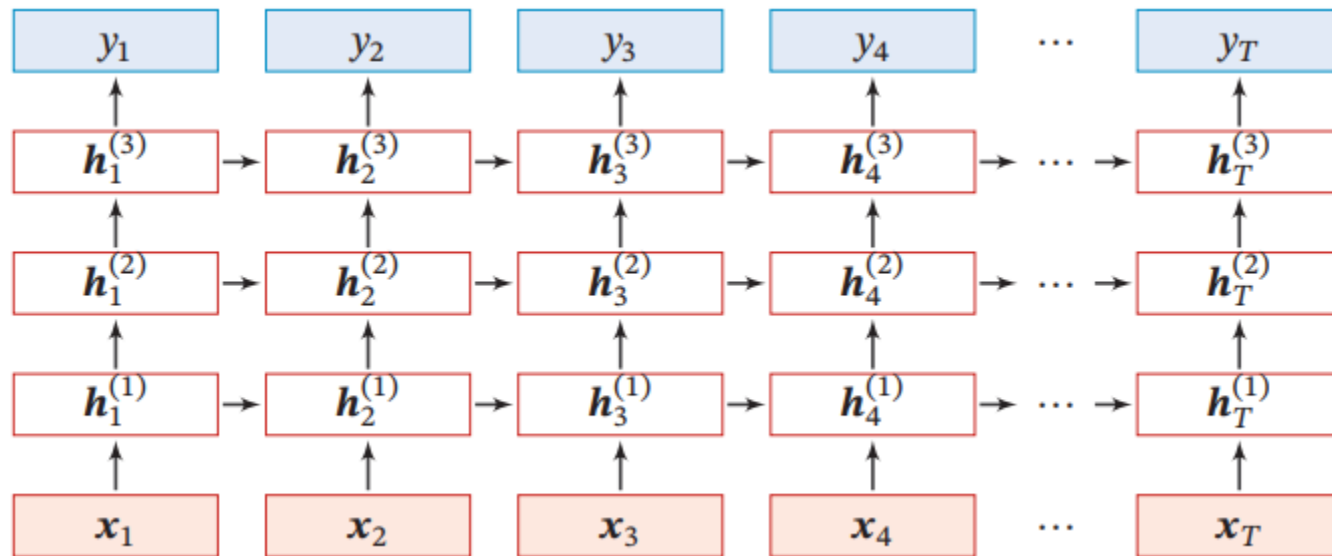
9.4 深层循环神经网络

- 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



9.4 深层循环神经网络

堆叠循环神经网络 (Stacked Recurrent Neural Network, SRNN)



第 l 层网络的输入是第 $l - 1$ 层网络的输出. 我们定义 $\mathbf{h}_t^{(l)}$ 为在时刻 t 时第 l 层的隐状态

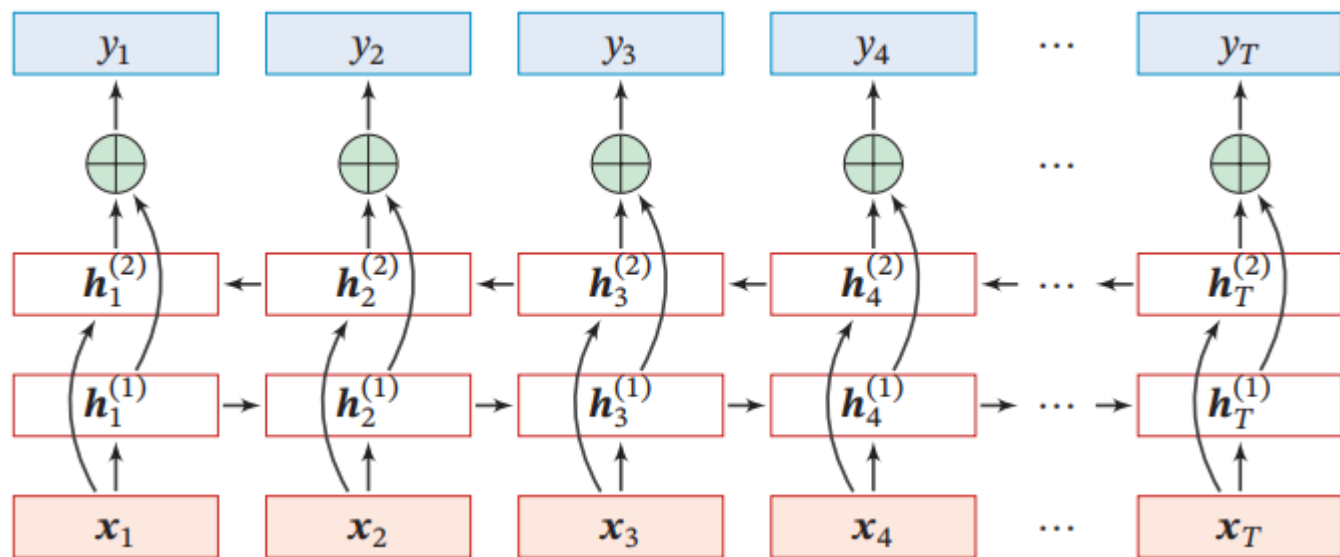
$$\mathbf{h}_t^{(l)} = f(\mathbf{U}^{(l)} \mathbf{h}_{t-1}^{(l)} + \mathbf{W}^{(l)} \mathbf{h}_t^{(l-1)} + \mathbf{b}^{(l)}),$$

其中 $\mathbf{U}^{(l)}$ 、 $\mathbf{W}^{(l)}$ 和 $\mathbf{b}^{(l)}$ 为权重矩阵和偏置向量, $\mathbf{h}_t^{(0)} = \mathbf{x}_t$.



9.4 深层循环神经网络

双向循环神经网络(Bidirectional Recurrent Neural Network)由两层循环神经网络组成，它们的输入相同，只是信息传递的方向不同



假设第1层按时间顺序，第2层按时间逆序，在时刻 t 时的隐状态定义为 $h_t^{(1)}$ 和 $h_t^{(2)}$ ，则：

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

$$h_t^{(2)} = f(U^{(2)}h_{t+1}^{(2)} + W^{(2)}x_t + b^{(2)}),$$

$$h_t = h_t^{(1)} \oplus h_t^{(2)},$$



主要参考文献

1. 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.
2. Understanding LSTM Networks <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>
3. Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.
4. Jozefowicz R, Zaremba W, Sutskever I. An empirical exploration of recurrent network architectures[C]//International conference on machine learning. 2015: 2342-2350.
5. 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.
6. LSTM Forward and Backward Pass <http://arunmallya.github.io/writeups/nn/lstm/index.html#/>