序列建模:循环和递归网络

Sequence Modeling: Recurrent and Recursive Nets

序列建模：循环和递归网络

主要内容

01 循环神经网络

02 深度循环神经网络

03 双向循环神经网络

04 LSTM及其变种

05 递归神经网络

06 基于编码-解码的序列到序列架构

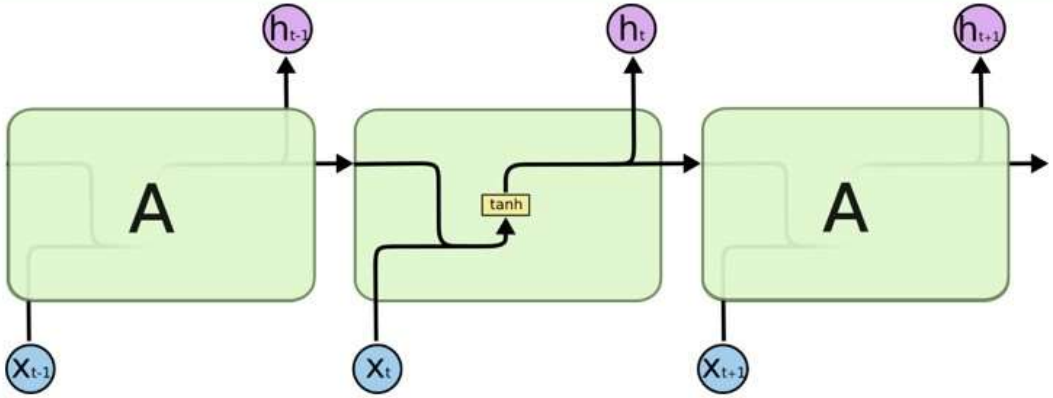
2

PART FOUR

LSTM及其变种

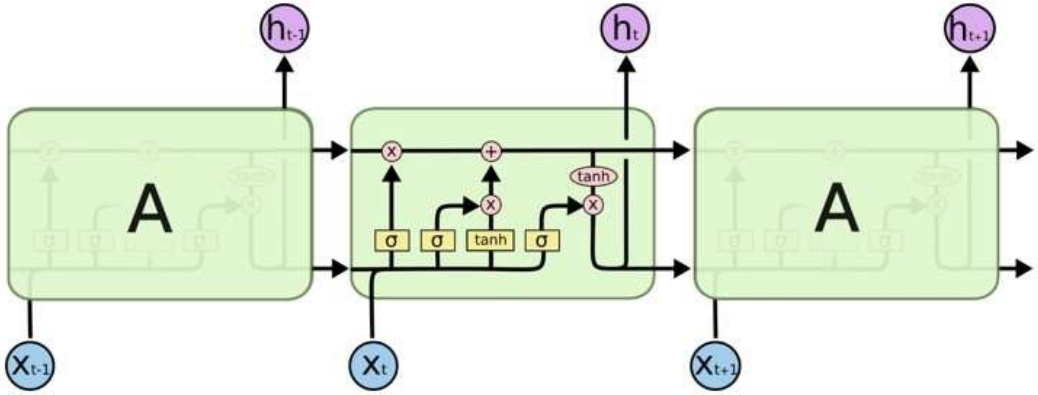
二、长短时记忆网络

RNN



LSTM

Long Short-Term Memory



4

填空题 3分

长短时记忆网络 LSTM 有三个门控单元，
分别是
[填空1] [填空2] [填空3]

单选题 1.5分

下列关于长短时记忆网络 LSTM 和循环神经网络 RNN 的关系描述正确的是（ ）

A LSTM 是简化版的 RNN

B LSTM 是双向的 RNN

C LSTM 是多层的 RNN

D LSTM 是 RNN 的扩展，其通过特殊的结构设计来避免长期依赖问题

6

单选题 1分

给定一个长度为 n 的不完整单词序列，希望预测下一个字母是什么。
比如输入是“softwar”（7 个字母组成），希望预测第 8 个字母是什么。
下面哪种神经网络结构适用于解决这个任务？（）

A 变分自编码网络

B 循环神经网络

C 卷积神经网络

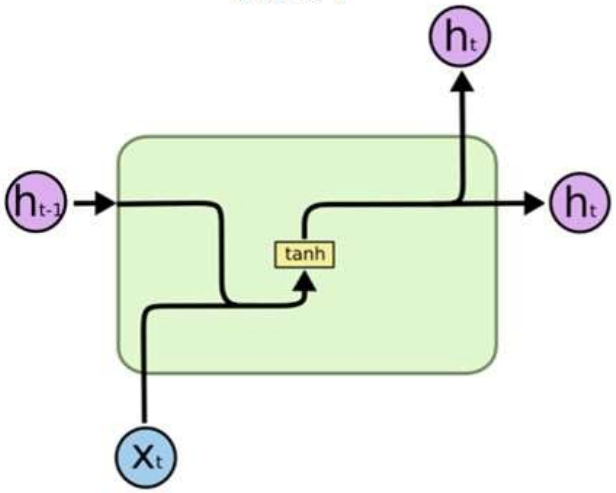
D 深度信念网络

7

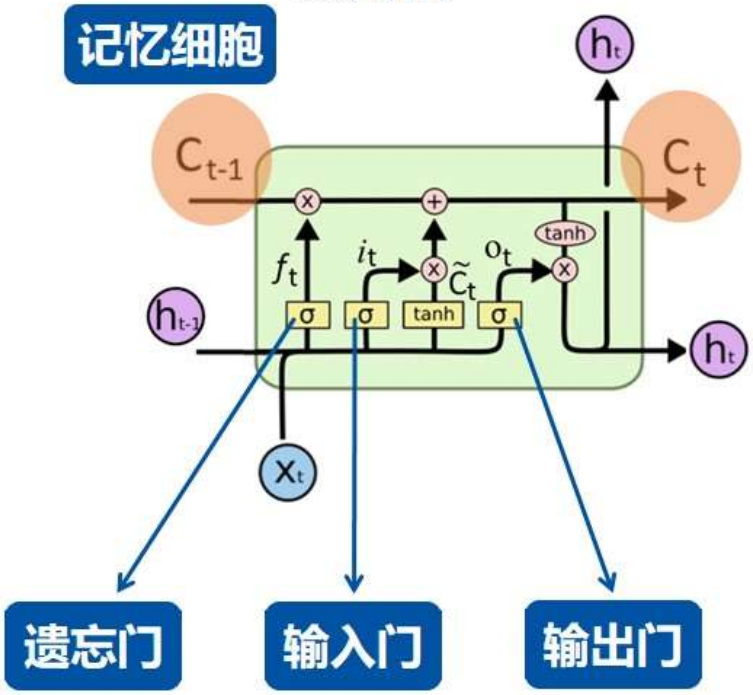
二、长短时记忆网络

好记性不如烂笔头

RNN



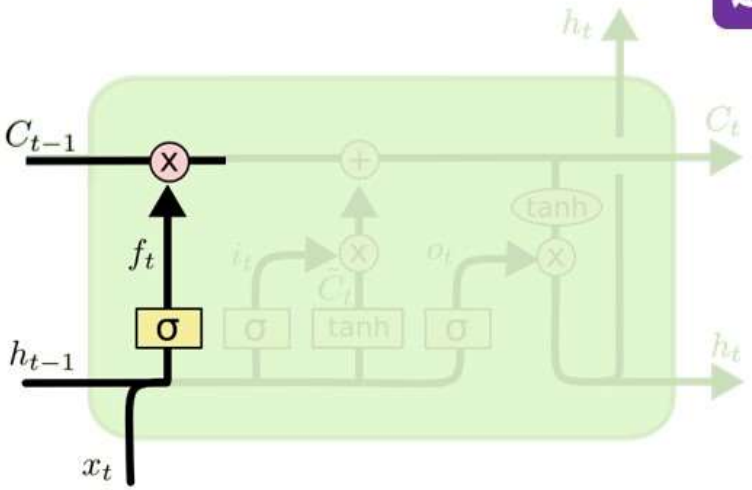
LSTM

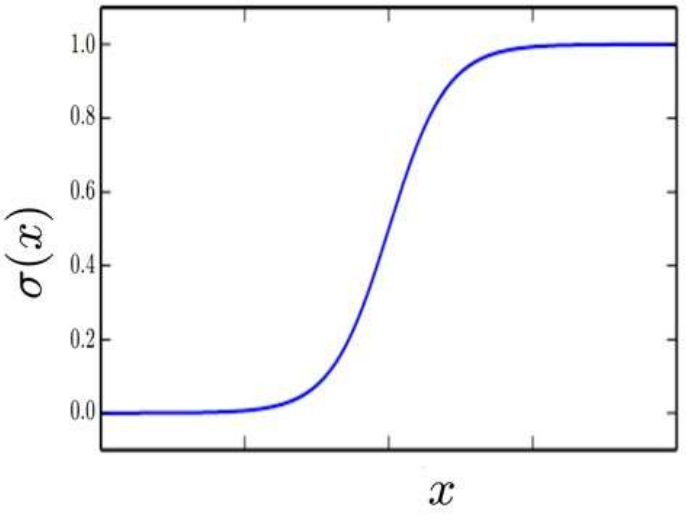


8

二、长短时记忆网络

遗忘门





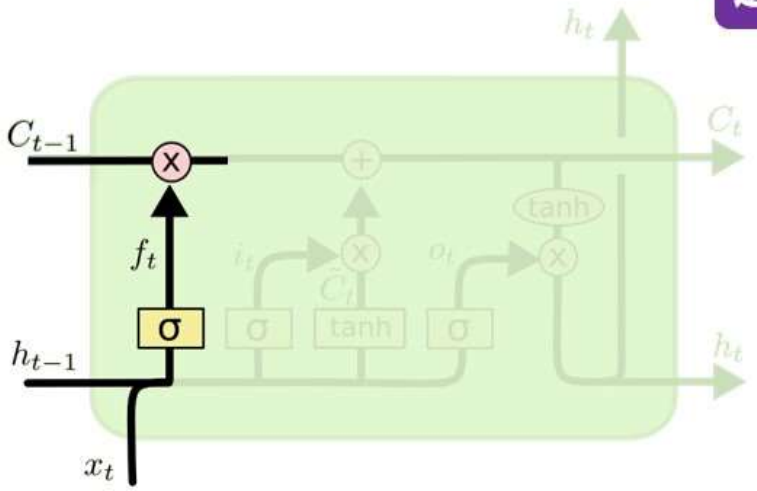
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

9

二、长短时记忆网络

遗忘门

决定本子上记的东西中，哪一部分要擦除，以腾出空间记新的事



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

$$\begin{bmatrix} 5 & 9 & 3 & 20 \end{bmatrix} \; C_{t-1}$$

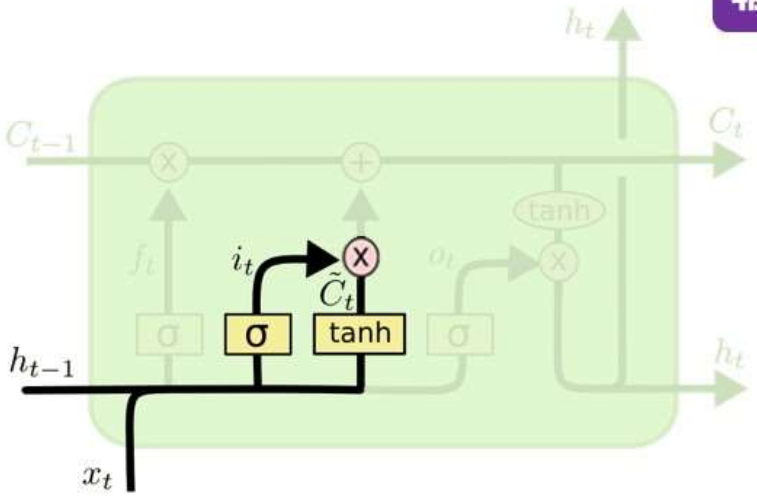
$$\begin{bmatrix} 0 & 1 & 0 & 0.2 \end{bmatrix} \; f_t$$

↓

$$\begin{bmatrix} 0 & 9 & 0 & 4 \end{bmatrix} \; f_t * C_{t-1}$$

10

二、长短时记忆网络



输入门

决定当前信息的哪一部分应该记到本子上

$$\begin{bmatrix} 15 & 12 & 20 & 6 \end{bmatrix} \tilde{C}_t$$

$$\begin{bmatrix} 0 & 1 & 0 & 0.5 \end{bmatrix} i_t$$

↓

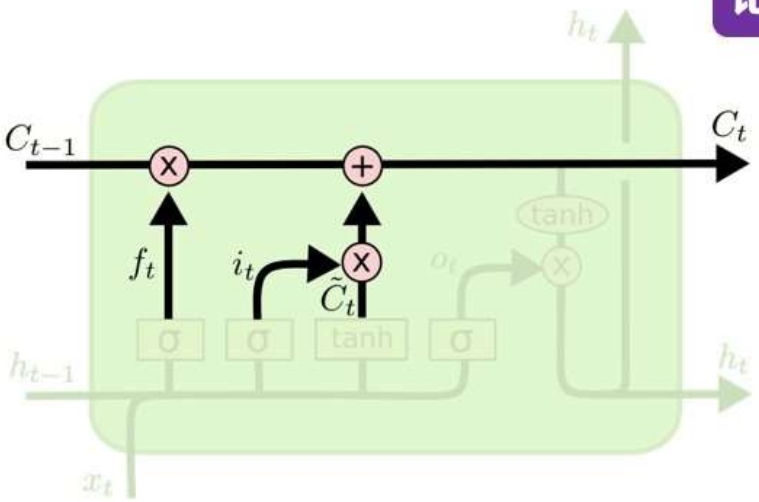
$$\begin{bmatrix} 0 & 12 & 0 & 3 \end{bmatrix} i_t * \tilde{C}_t$$

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

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

11

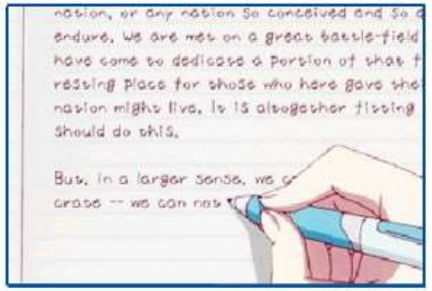
二、长短时记忆网络



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

记忆更新

新的内容被记录到小本子上



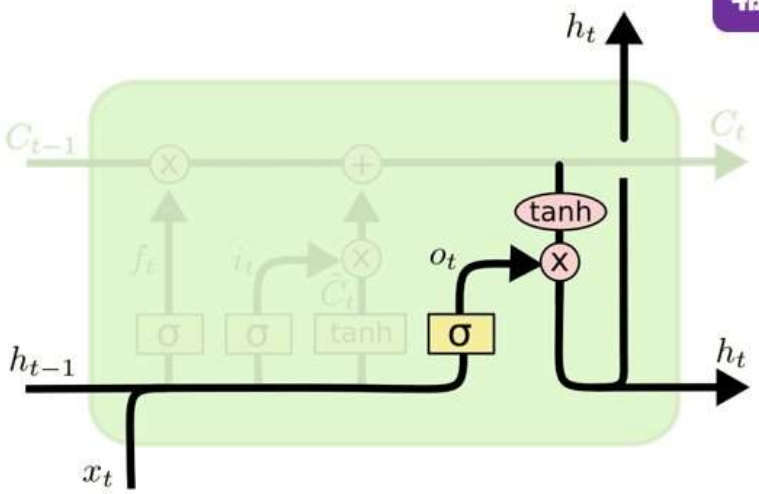
$$\begin{bmatrix} 0 & 9 & 0 & 4 \end{bmatrix} f_t * C_{t-1}$$

$$\begin{bmatrix} 0 & 12 & 0 & 3 \end{bmatrix} i_t * \tilde{C}_t$$

$$\downarrow$$

$$\begin{bmatrix} 0 & 21 & 0 & 7 \end{bmatrix} C_t$$

二、长短时记忆网络



输出门

决定小本子上的哪一部分内容
格外重要，需要牢记在心。
复习之，并记到脑子里。

$$\begin{bmatrix} 0 & 21 & 0 & 7 \end{bmatrix} \tanh(C_t)$$

$$\begin{bmatrix} 0 & 1 & 0 & 0.5 \end{bmatrix} o_t$$

↓

$$\begin{bmatrix} 0 & 21 & 0 & 3.5 \end{bmatrix} h_t$$

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

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

13

二、长短时记忆网络

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

计算：

$$\delta_{t-1} = \frac{\partial C_T}{\partial C_{t-1}} = \frac{\partial C_T}{\partial C_t}\frac{\partial C_t}{\partial C_{t-1}} = \delta_t \frac{\partial C_t}{\partial C_{t-1}} = \delta_t(f_t + \cdots)$$

当 f_t=1 时，即使其余项很小，梯度仍然可以很好传到上一个时刻，此时即使层数较深也不会发生梯度消失的问题；

14

LSTM

LSTM的总结

1. **新记忆产生：**使用输入词x_t和过去隐层状态h_{t-1}来产生新的记忆$\tilde{C}_t$。

2. **输入门：**在产生新记忆之前，我们需要判定一下我们当前看到的新词到底重不重要，这就是输入门的作用。输入门根据输入词和过去隐层状态共同判定输入值是否值得保留，从而判定它以何种程度参与生成新的记忆(或者说对新的记忆做一个约束)。因此，它可以作为输入信息更新的一个指标。

3. **遗忘门：**这个门和输入门很类似。但是它不能决定输入词有效，它能对过去记忆单元是否对当前记忆单元的计算有用做出评估。

4. **最终记忆产生：**这个阶段会根据遗忘门的作用结果，合理地忘记部分过去的记忆C_{t-1}。再根据输入门i_t的作用结果，产生新记忆$\tilde{C}_t$。它将这两个结果相加融合起来产生了最终的记忆C_t。

5. **输出门：**它的目的是从隐层状态分离最终的记忆。最终记忆C_t包含了大量不必需要保存在隐层状态的信息，这个门限能够评估关于记忆C_t哪部分需要显示在隐层状态h_t中。用于评估这部分信息的中间信号叫做o_t，它和$tanh(C_t)$的点乘组成最后的h_t。

15

LSTM变体

LSTM变体

16

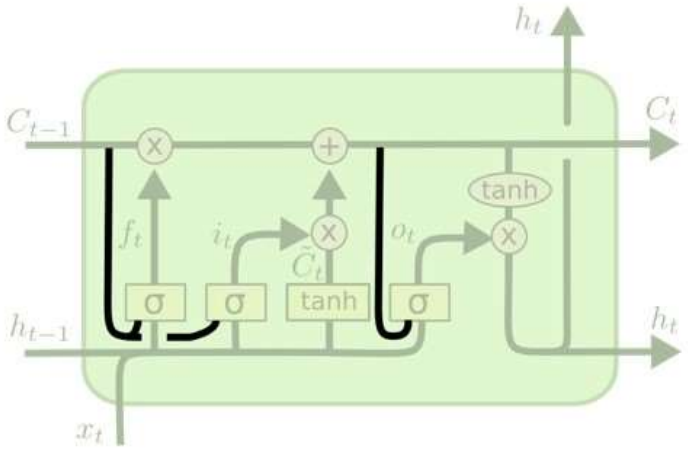
LSTM

变体

变体1

① 增加“peephole connection”。

② 让“门层”也接受细胞状态的输入。



$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] + b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] + b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] + b_o\right)$$

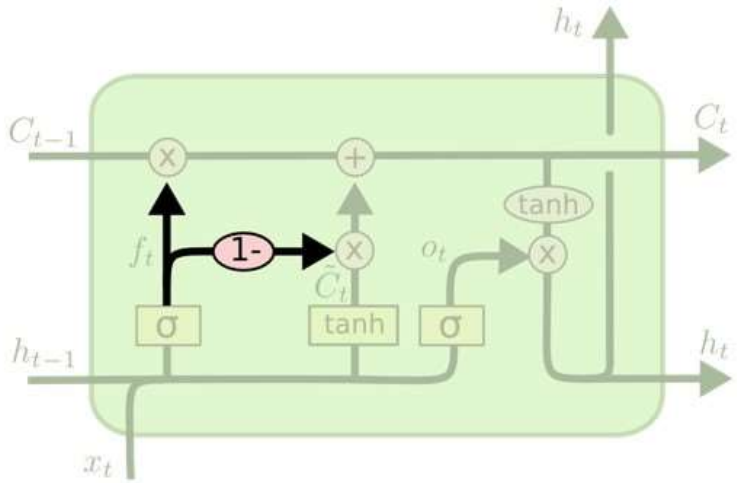
LSTM

变体

变体2

① 通过使用coupled忘记和输入门。

② 不同于之前是分开确定需要忘记和添加的信息，这里是一同做出决定。



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

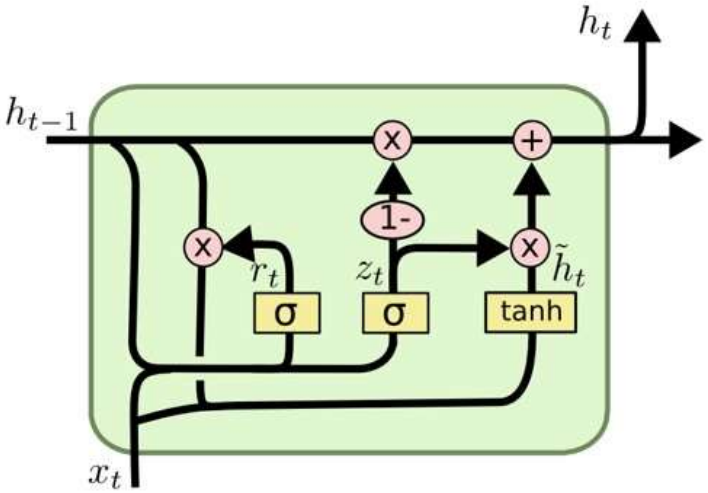
LSTM

变体

- 变体3 -- Gated Recurrent Unit (GRU)

① 将忘记门和输入门合成了一个单一的**更新门**。

② 混合了细胞状态和隐藏状态，和其他一些改动。最终的模型比标准的 LSTM 模型要简单，是非常流行的变体。



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

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

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

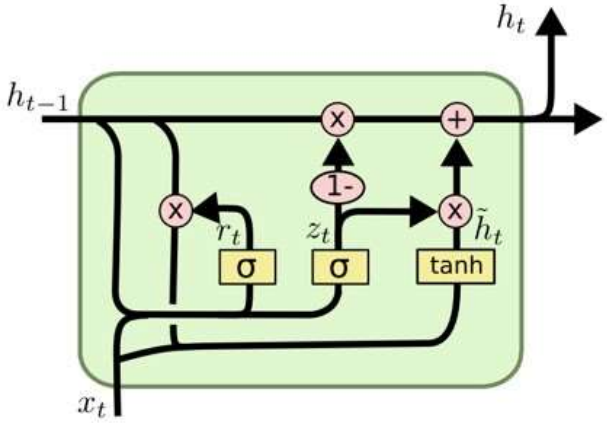
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

LSTM

变体

GRU的再次理解

- **GRU**模型如下，它只有两个门了，分别为**更新门**和**重置门**，即图中的z_t和r_t。**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，更新门的值越小说明前一时刻的状态信息带入越多。**重置门**用于控制忽略前一时刻的状态信息的程度，重置门的值越小说明忽略得越多。



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

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

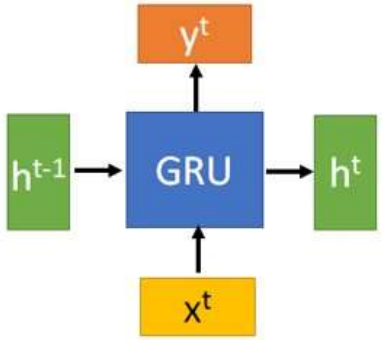
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

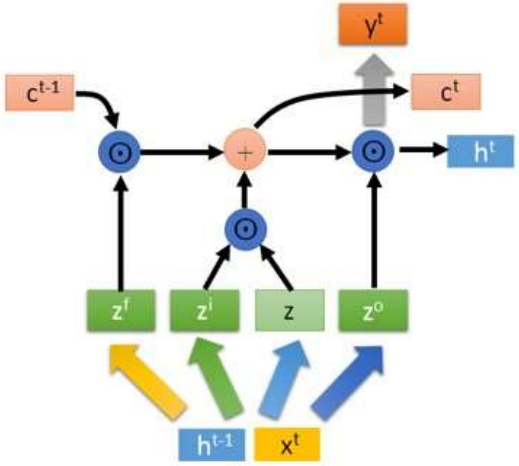
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

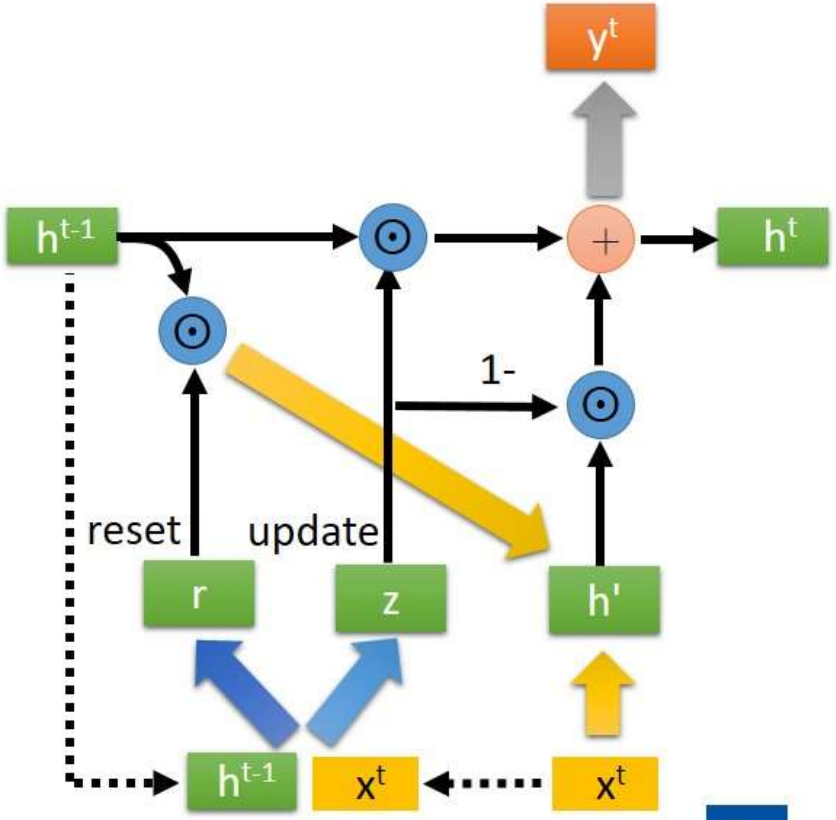
LSTM

变体GRU

$$h^t = z \odot h^{t-1} + (1-z) \odot h'$$







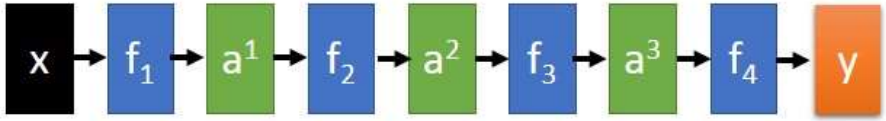
21

Highway Network

Highway Network
和
Residual Network
(ResNet)

一、循环神经网络的原理

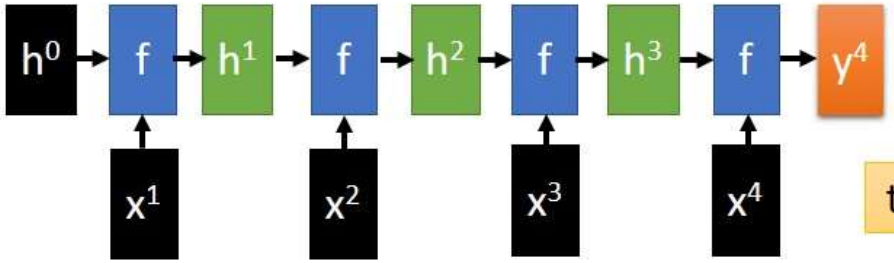
1. Feedforward network does not have input at each step
2. Feedforward network has different parameters for each layer



$$a^t = f_l(a^{t-1}) = \sigma(W^t a^{t-1} + b^t)$$

t is layer

Feedforward v.s. Recurrent



t is time step

$$h^t = f(h^{t-1}, x^t) = \sigma\left(W^h h^{t-1} + W^i x^t + b^i\right)$$

Slide credit: Hung-yi Lee

Applying gated structure in feedforward network

23

GRU → Highway Network

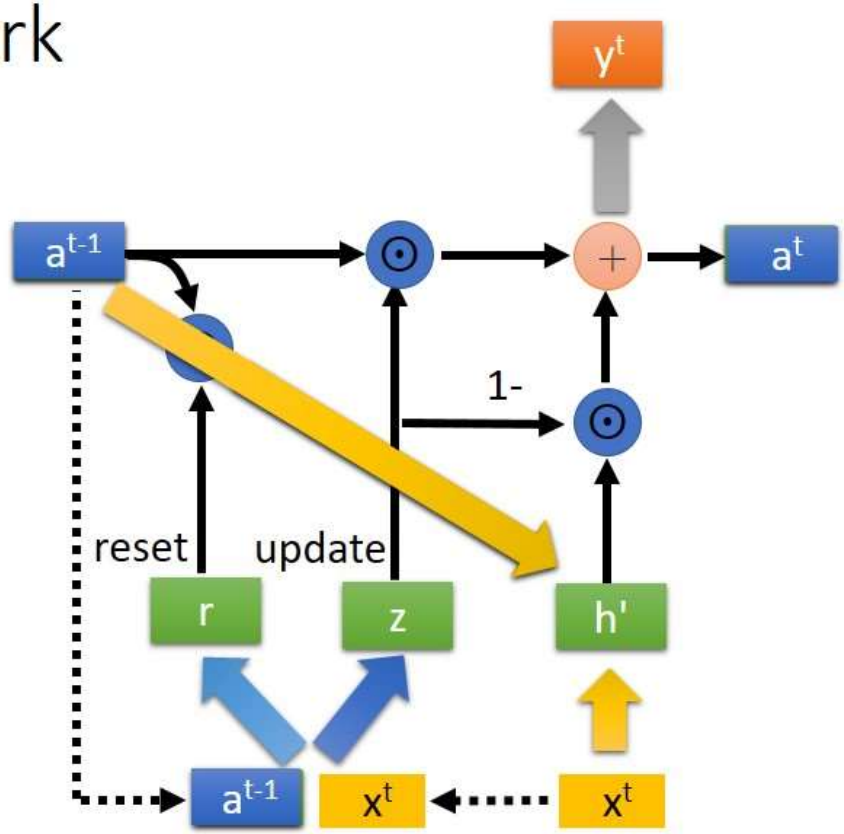
No input x^t at each step

No output y^t at each step

a^{t-1} is the output of the (t-1)-th layer

a^t is the output of the t-th layer

No reset gate



Slide credit: Hung-yi Lee

用 H 来表示一个非线性转换，参数为 W_H ，当输入为 x 时，输出为

$$y = H(x, W_H)$$

上面的公式就是最普通的一层神经网络的表示。而Highway Network则定义了两个非线性变换 $T(x, W_T)$ 和 $C(x, W_C)$ ，这两个作为系数，网络输出为

$$y = H(x, W_H) * T(x, W_T) + x * C(x, W_C)$$

文中将 T 作为 transform gate，表示有多少转换后的信息通过， C 为carry gate，表示有多少原始信息通过，为了简单，设置 $C = 1 - T$ ，那么有

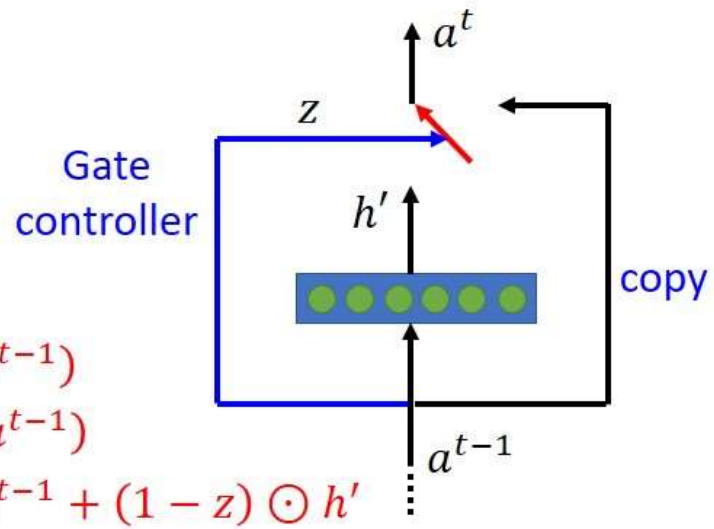
$$y = H(x, W_H) * T(x, W_T) + x * (1 - T(x, W_T))$$

可以看出Highway Network和ResNet是比较相似的，都有两个分支进行合并。论文里 T 的具体定义为 $T(x) = \sigma(W_T^{\mathbf{T}} x + b_T)$ ，σ 是sigmoid函数。

Highway Network

$$a^t = a^{t-1} + h'$$

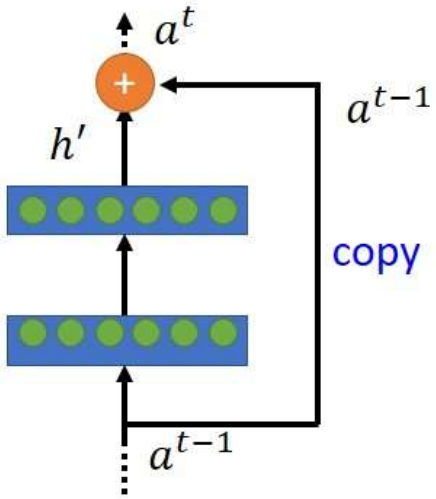
• Highway Network



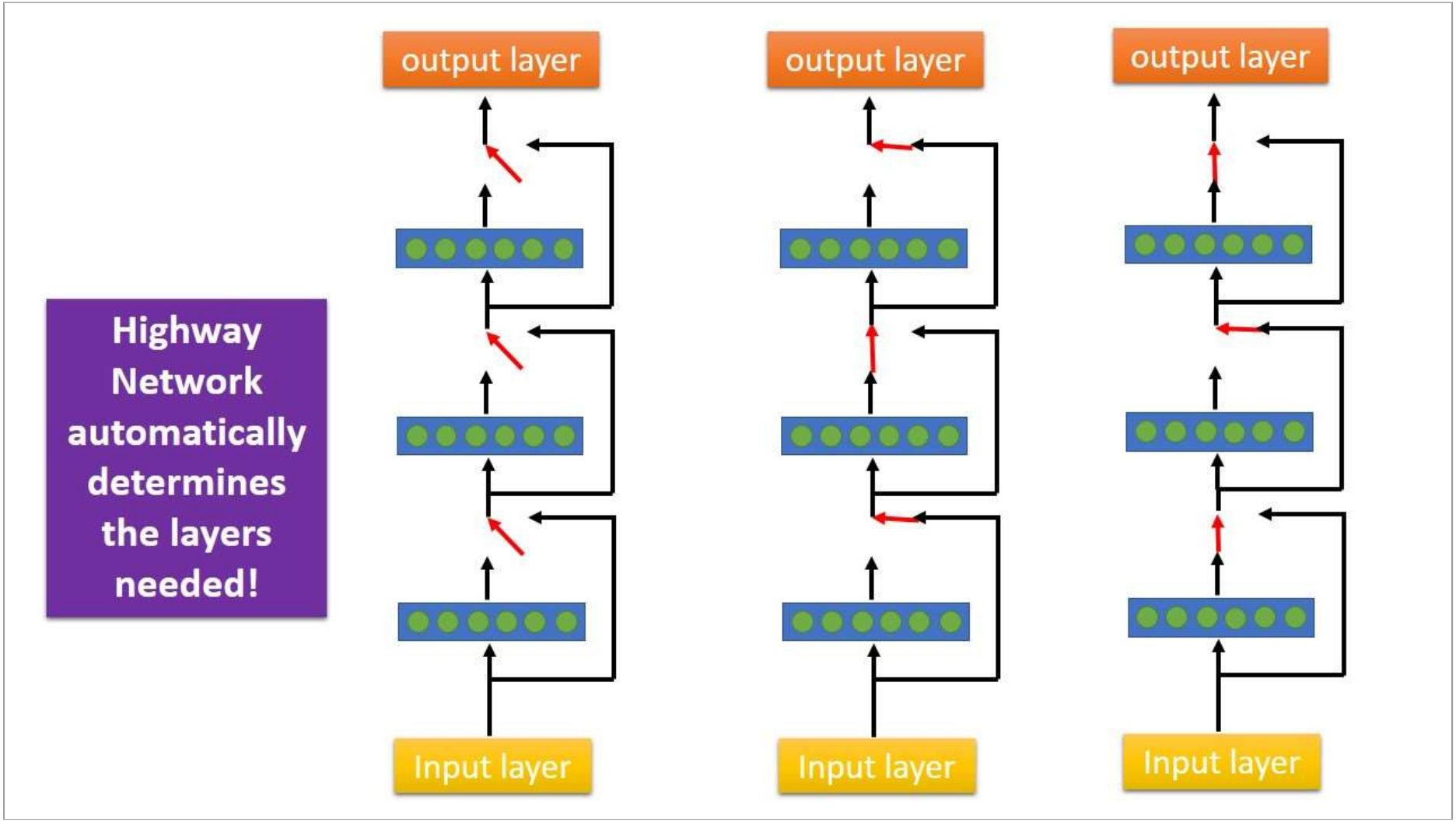
$$h' = \sigma(W a^{t-1})$$
$$z = \sigma(W' a^{t-1})$$
$$a^t = z \odot a^{t-1} + (1 - z) \odot h'$$

Training Very Deep Networks
https://arxiv.org/pdf/1507.06228v2.pdf

• Residual Network

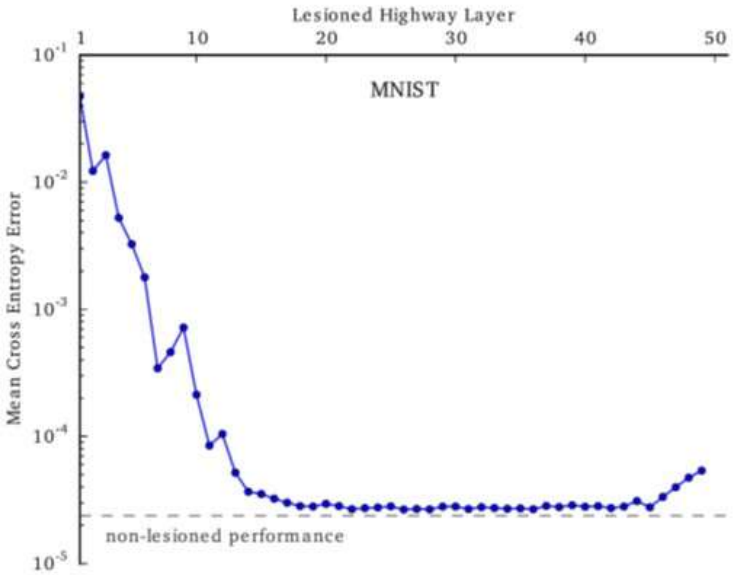


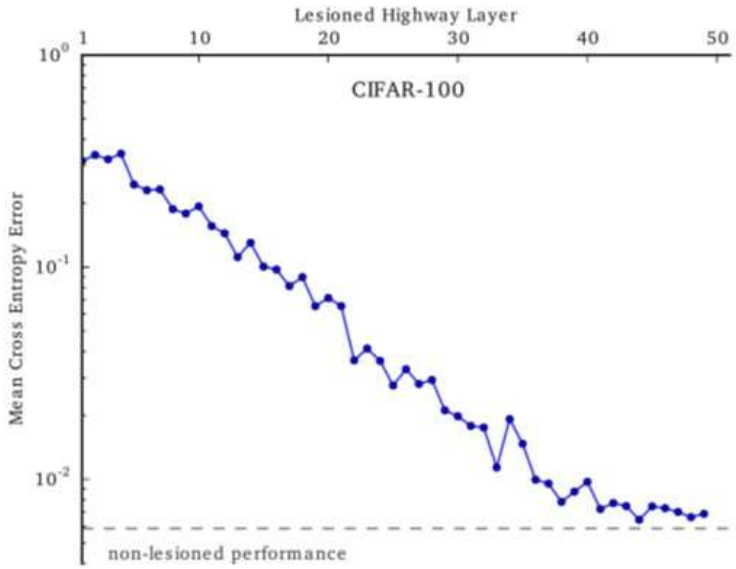
Deep Residual Learning for Image Recognition
http://arxiv.org/abs/1512.03385


Highway Network automatically determines the layers needed!
output layer
output layer
output layer
Input layer
Input layer
Input layer

Highway Network

Highway Network



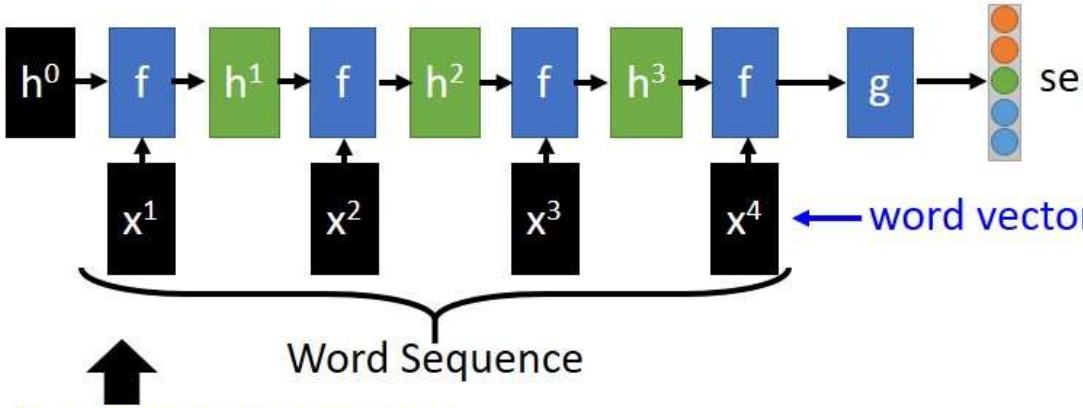


28

PART FIVE

递归神经网络
Recursive Structure

Application: Sentiment Analysis

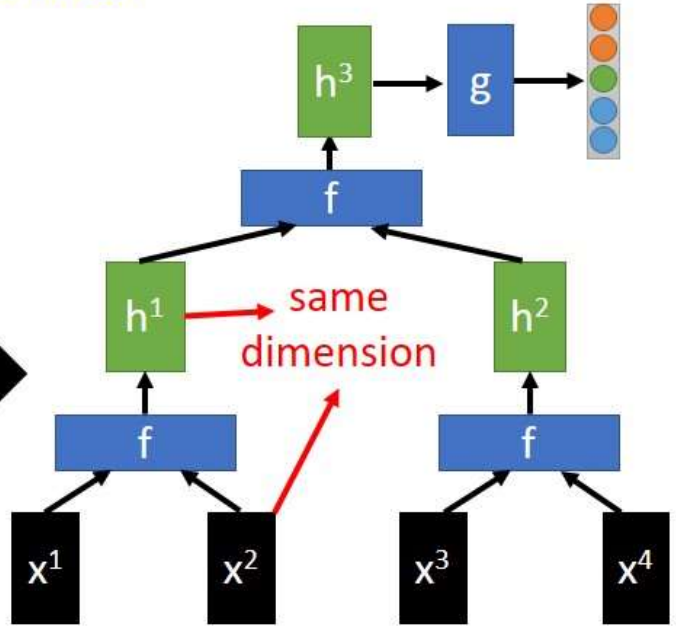


Recurrent Structure

Special case of recursive structure

Recursive Structure

How to stack function f is already determined



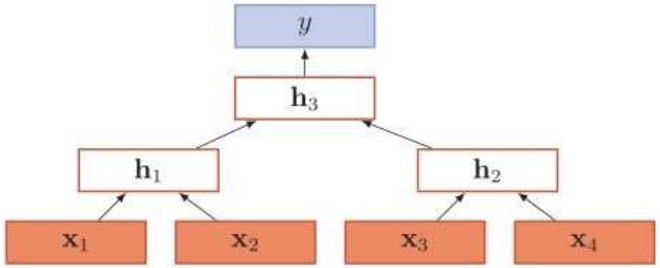
递归神经网络

- 递归神经网络实在一个有向图无循环图上共享一个组合函数
- Recursive Neural Network

$$\mathbf{h}_1 = f(W \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} + \mathbf{b}),$$

$$\mathbf{h}_2 = f(W \begin{bmatrix} \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix} + \mathbf{b}),$$

$$\mathbf{h}_3 = f(W \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \end{bmatrix} + \mathbf{b}),$$

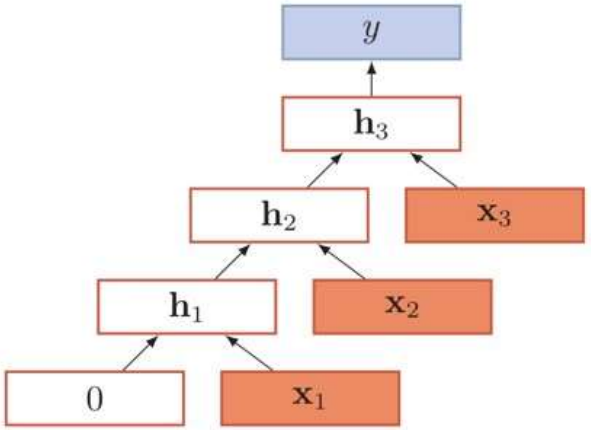


《神经网络与深度学习》

递归神经网络

- 退化为循环神经网络

$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{x}_t),$$



《神经网络与深度学习》


THANK YOU
Q&A