第 7 章 循环神经网络

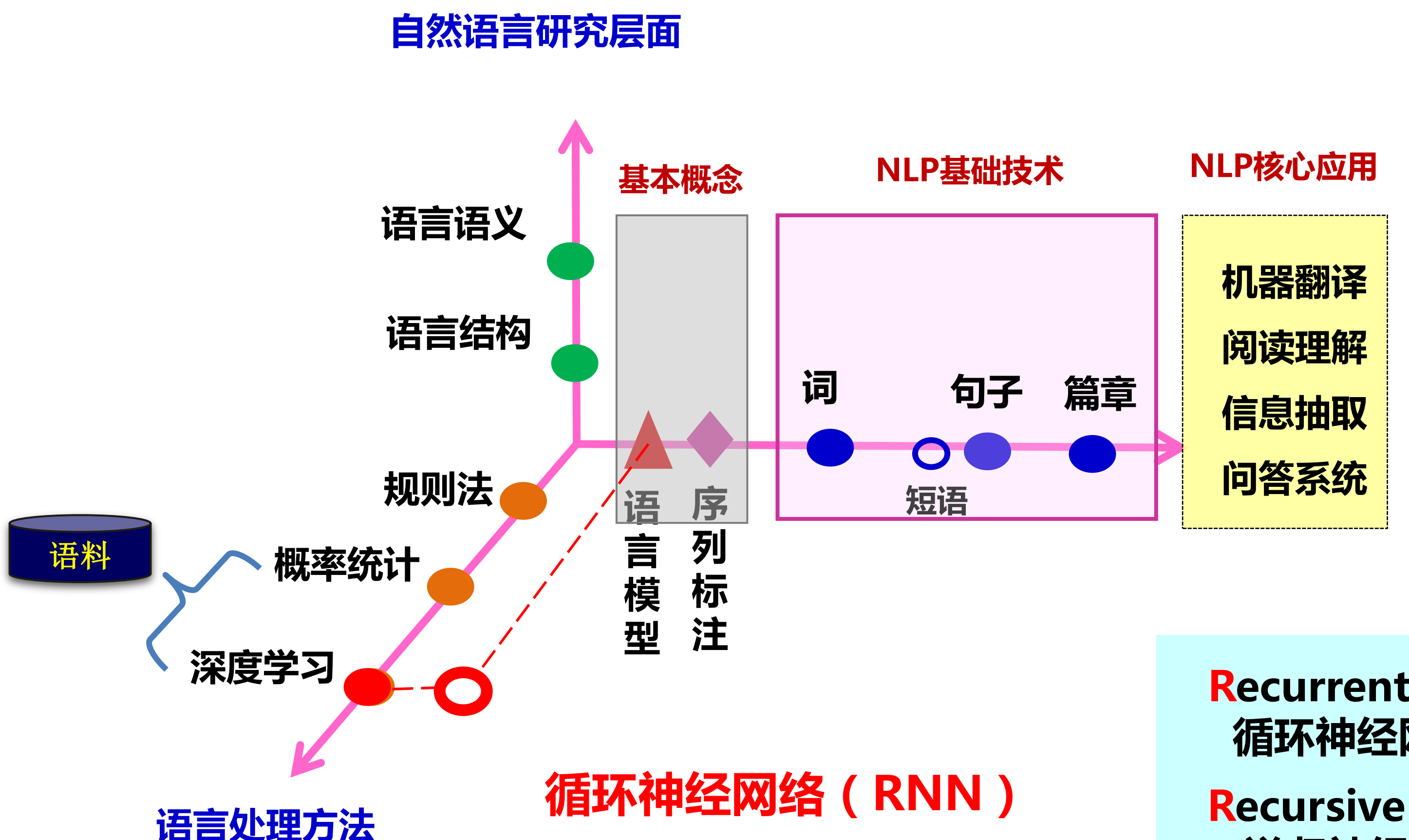
中科院信息工程研究所第二研究室

胡玥

huyue@iie.ac.cn

自然语言处理课程内容及安排

✧ **课程内容：**



Recurrent Neural Network
循环神经网络

Recursive Neural Network
递归神经网络

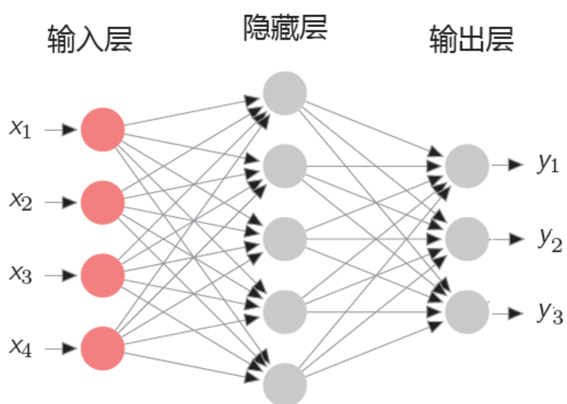
内 容 提 要

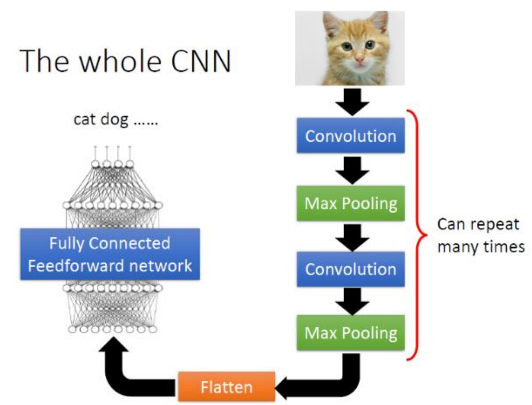
7.0　概述

问题引入：

1. DNN、CNN 输入、输出定长；处理输入、输出变长问题效率不高。
   而自然语言处理中的语句通常其长度不固定。

DNN

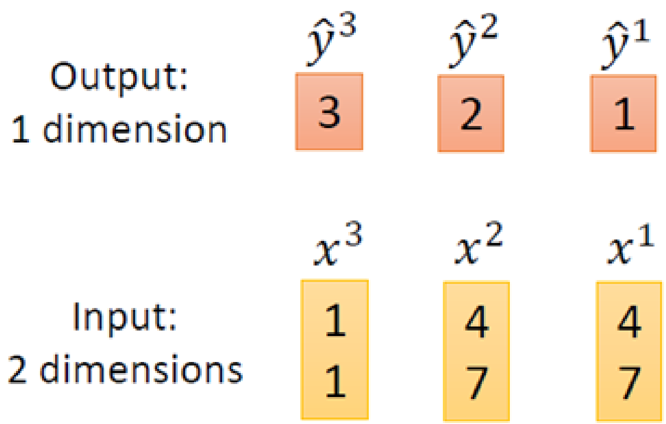


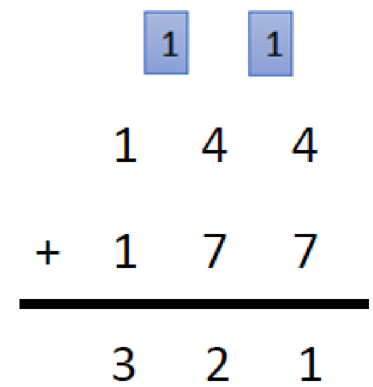


7.0 概述

2. 单一 DNN、CNN 无法处理时序相关序列问题

例如：





解决方法： **循环神经网络**

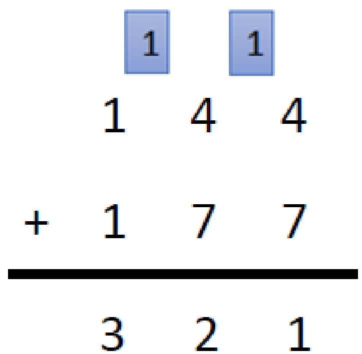
7.0　概述

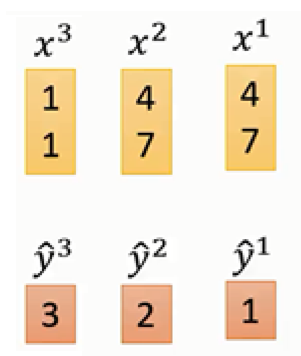
循环神经网络核心思想：

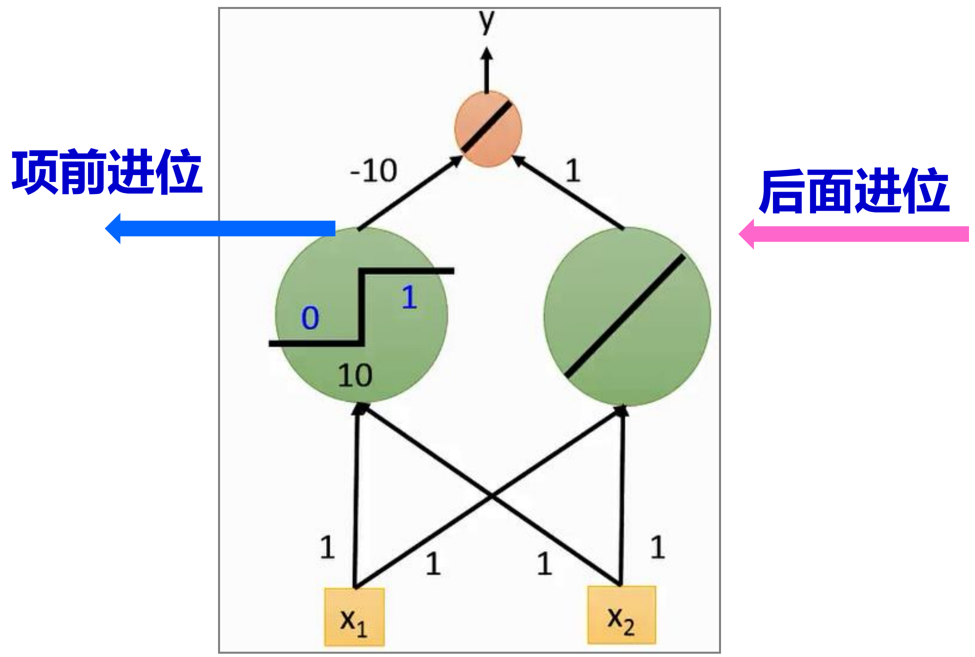
将处理问题在时序上分解为一系列相同的“单元”，单元的神经网络可以在时序上展开，且能将上一时刻的**结果传递给下一时刻**，整个网络按时间轴展开。即可变长。

7.0 概述

例如：加法问题



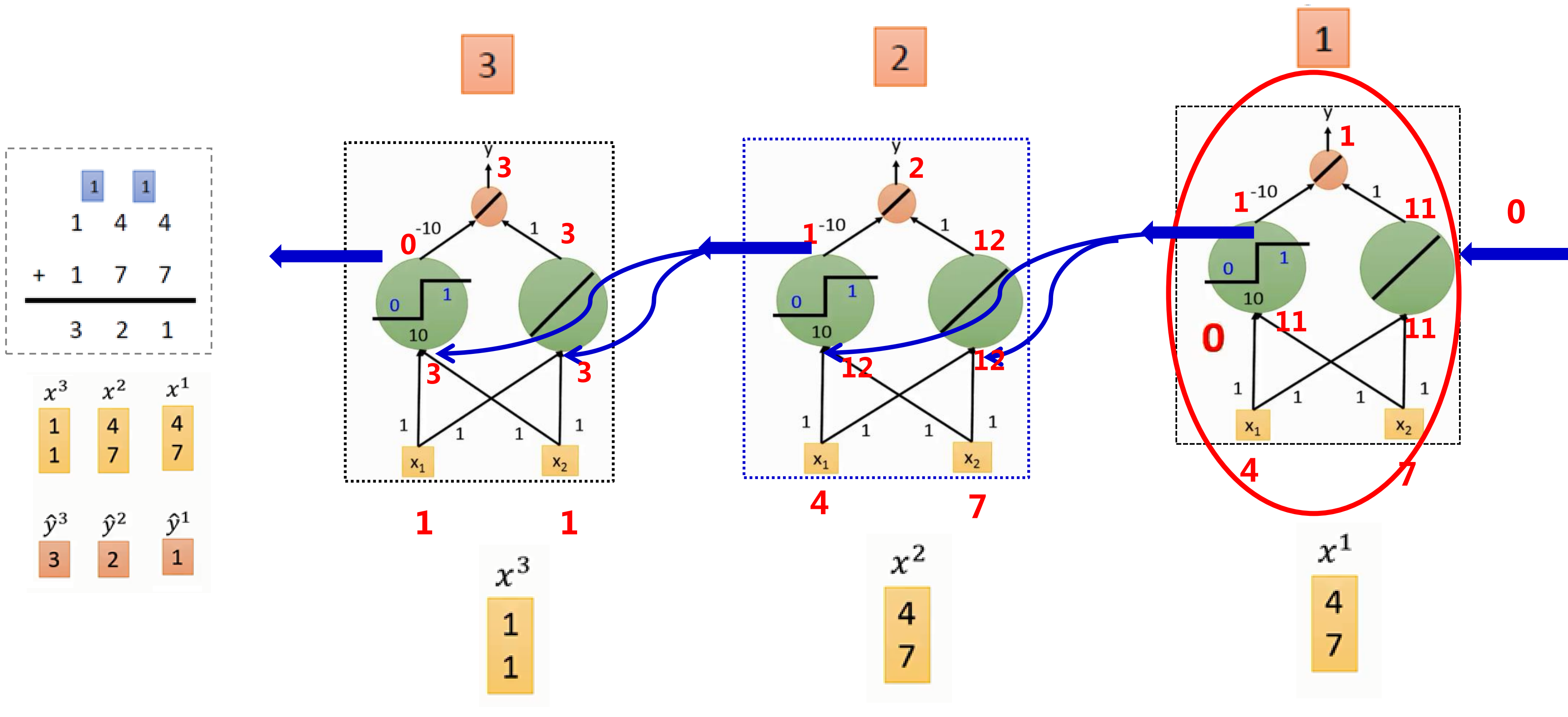




一位加法单元

7.0　概述

三位加法单元

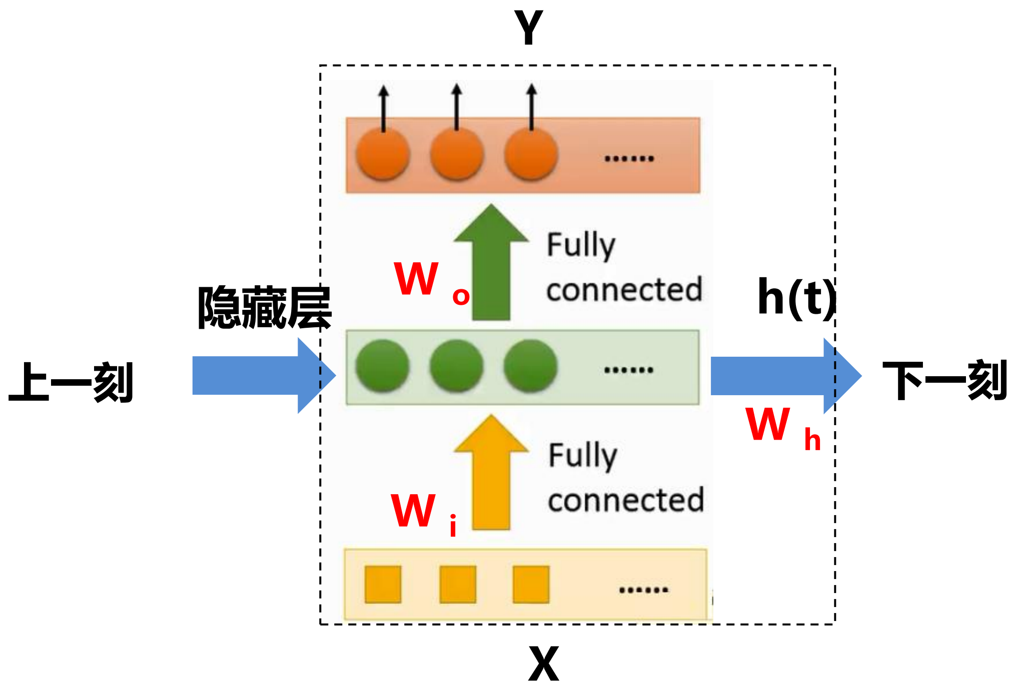


时序上展开

内 容 提 要

7.1 循环神经网络结构

RNN单元结构：



输入： X + 来自上时刻隐藏层

输出： Y + 给下时刻隐藏层

参数： W_i 、W_o 、W_h

信息传播：

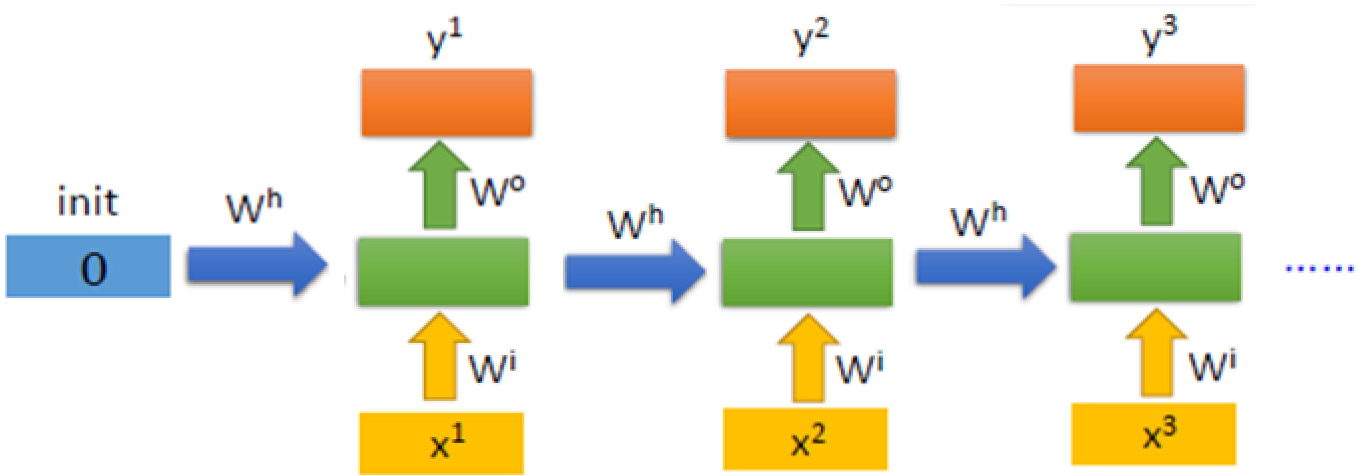
$$h(t) = \sigma(W_i X + W_h h(t-1) + b)$$

$$Y = softmax(W_o h(t))$$

7.1 循环神经网络结构

RNN网络结构（按时序展开）：

Input data: x^1 x^2 x^3 ...... x^N



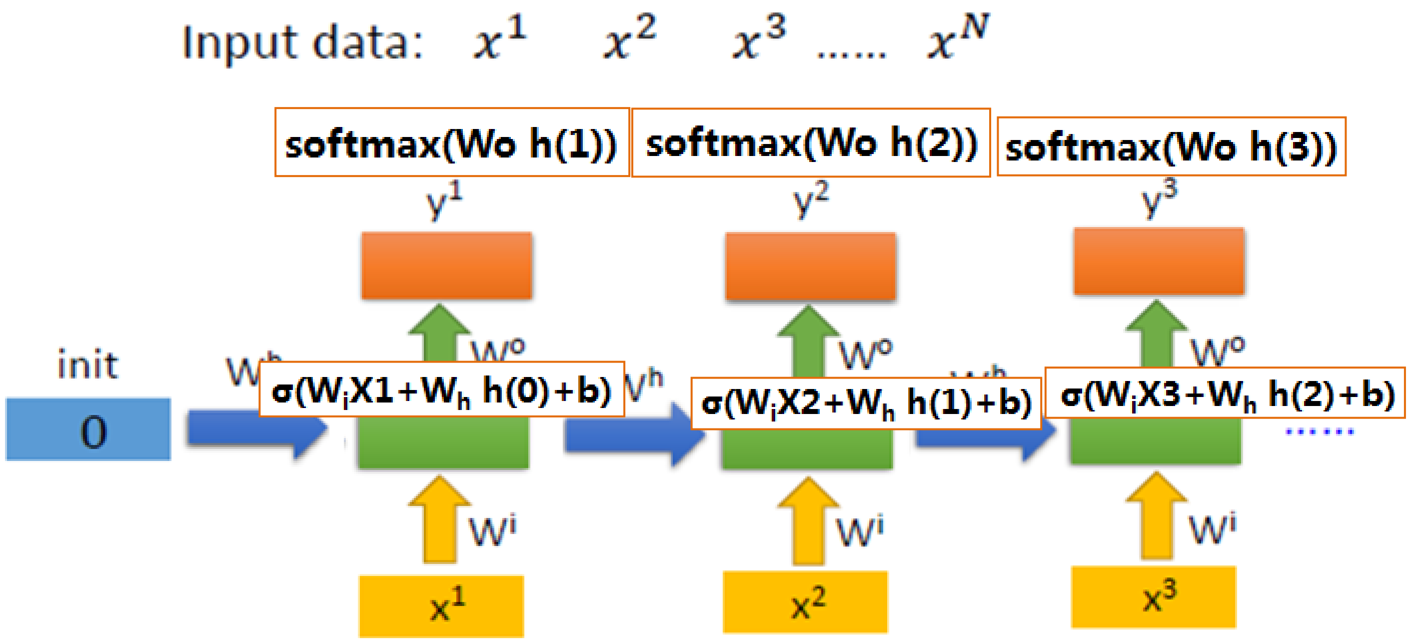
输入：X （$x^1x^2x^3$）

输出：Y （$y^1y^2y^3$）

参数： W_i 、W_o 、W_h

7.1 循环神经网络结构

RNN网络结构（按时序展开）：



输入：X（$x^1x^2x^3$）

输出：Y （$y^1y^2y^3$）

参数：W_i、W_o、W_h

信息传播：

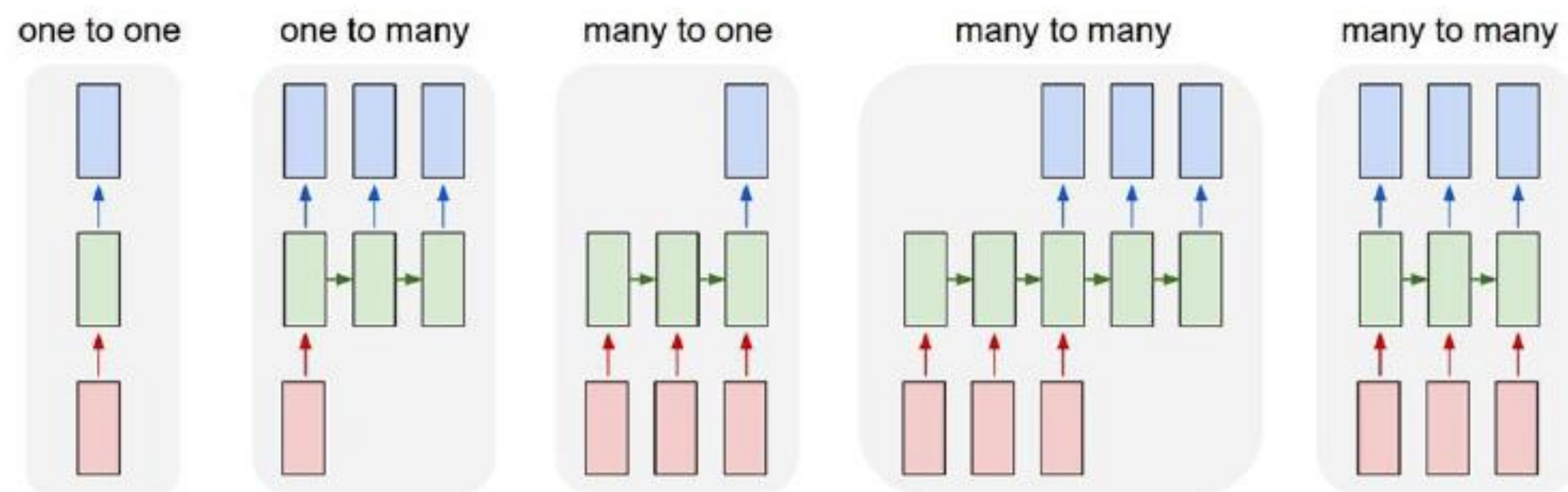
$$h(t) = \sigma(W_iX + W_h\ h(t-1) + b)$$

$$Y = softmax(W_o h(t))$$

7.1　循环神经网络结构

输入输出结构：

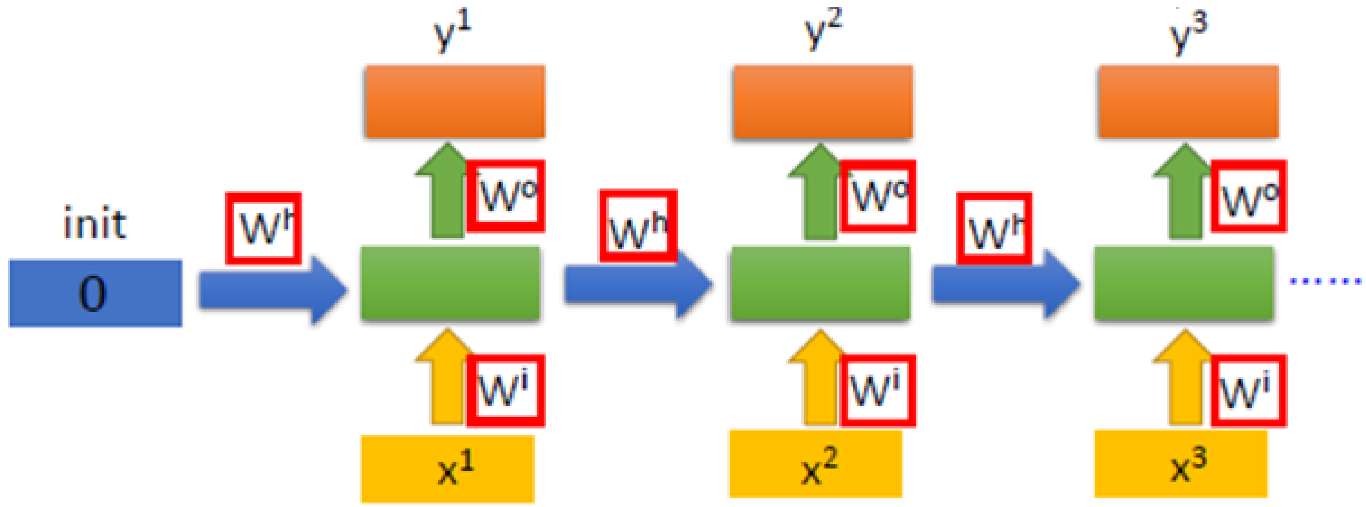
RNN输入和输出结构可以等长或不等长



内 容 提 要

7.2 循环神经网络训练

RNN参数



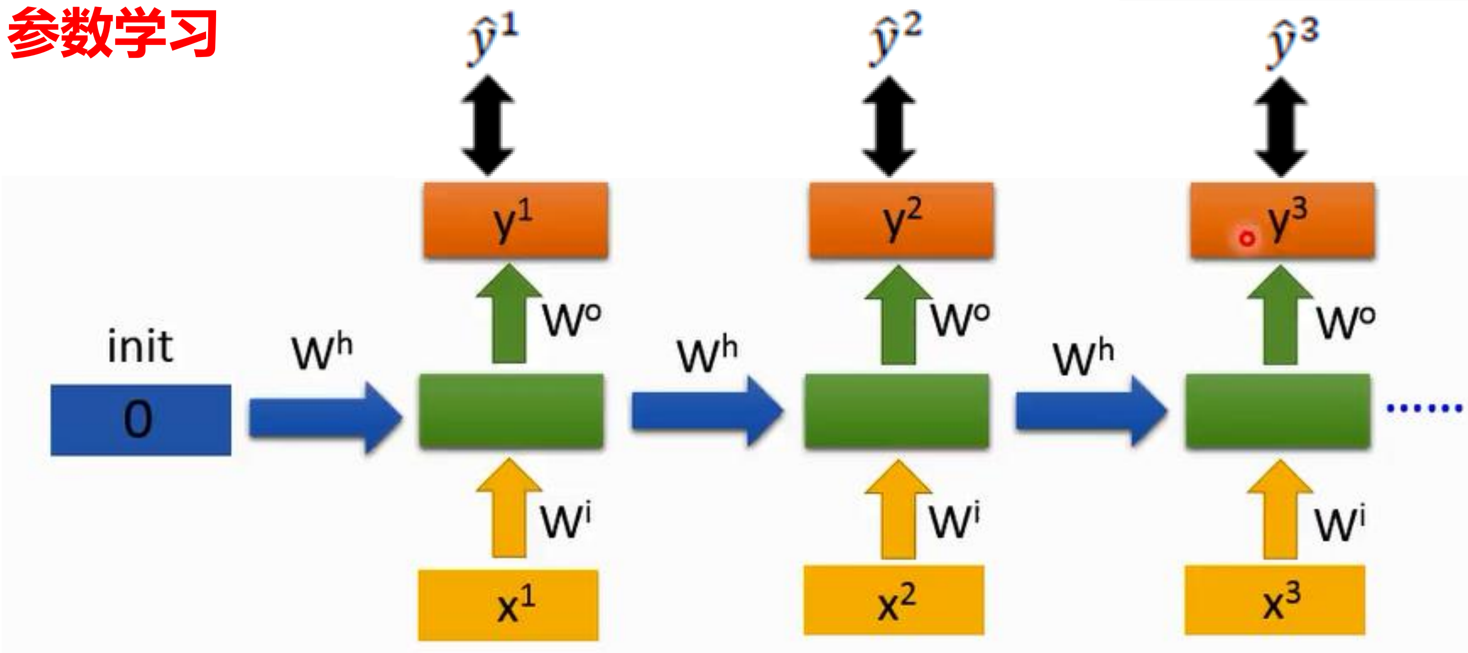
RNN参数： W_i、W_o、W_h、b

7.2 循环神经网络训练

$h(t) = \sigma(W_i X + W_h h(t-1) + b)$

$Y = \mathrm{softmax}(W_o h(t))$

参数学习



- 用 y^i 与 $\hat{y}^i$ 的误差定义

  损失函数： L(θ) 或 C(θ)

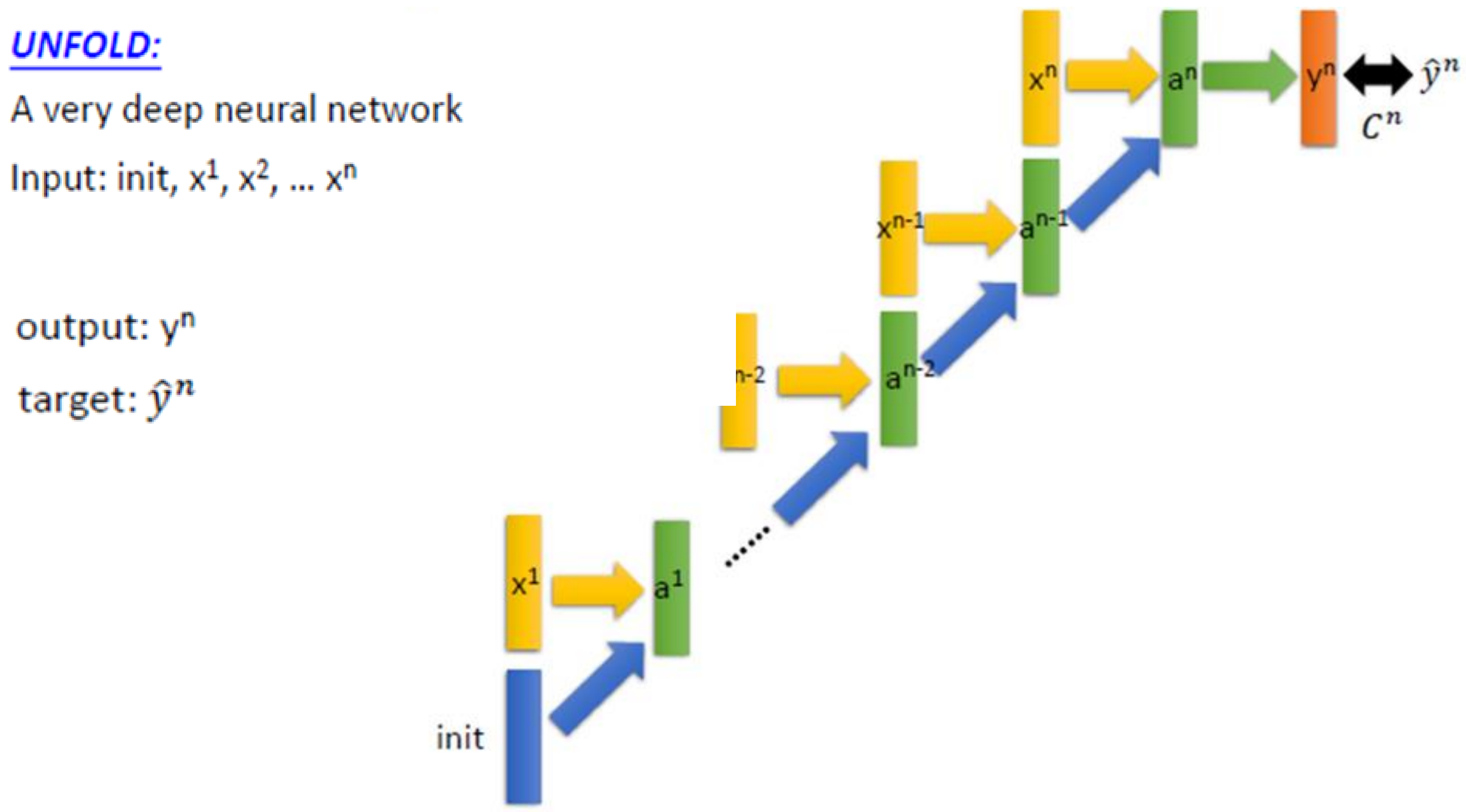
  $\Theta=\{W_i, W_o, W_h, b\}$

- 梯度下降法学习参数

  ➡ $w \leftarrow w - \eta \partial C^n / \partial w$

7.2 循环神经网络训练

BPTT（Backpropagation through time）



7.2 循环神经网络训练

BPTT

Backward Pass

$$\delta^L = \sigma'(z^L) \bullet \nabla C_x(y)$$

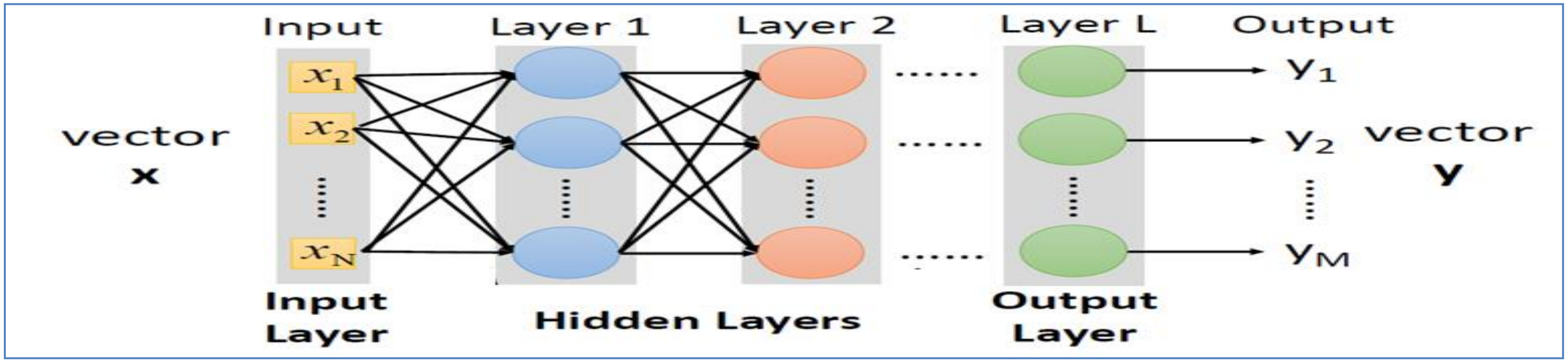
$$\delta^{L-1} = \sigma'(z^{L-1}) \bullet (W^L)^T \delta^L$$

……

$$\delta^l = \sigma'(z^l) \bullet (W^{l+1})^T \delta^{l+1}$$

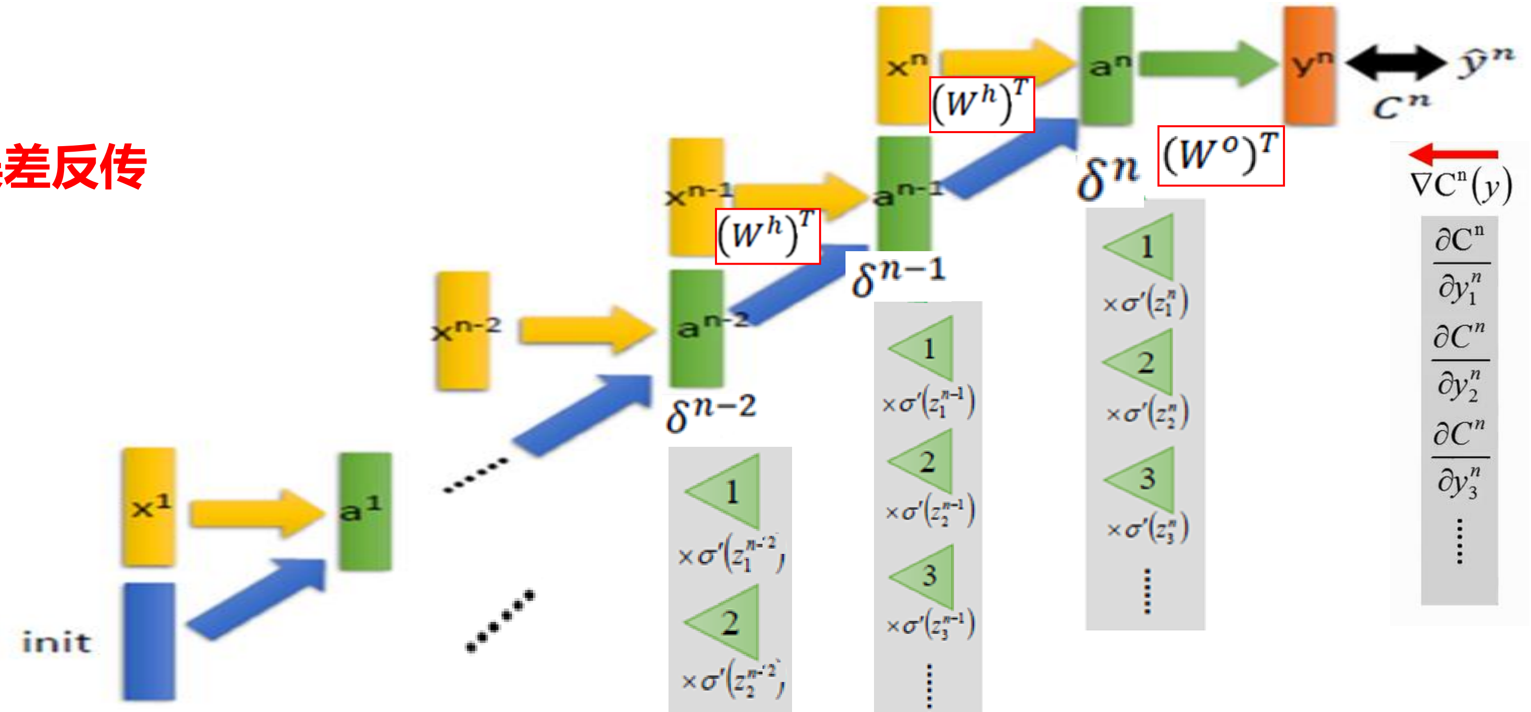
……

DNN



RNN

RNN 误差反传



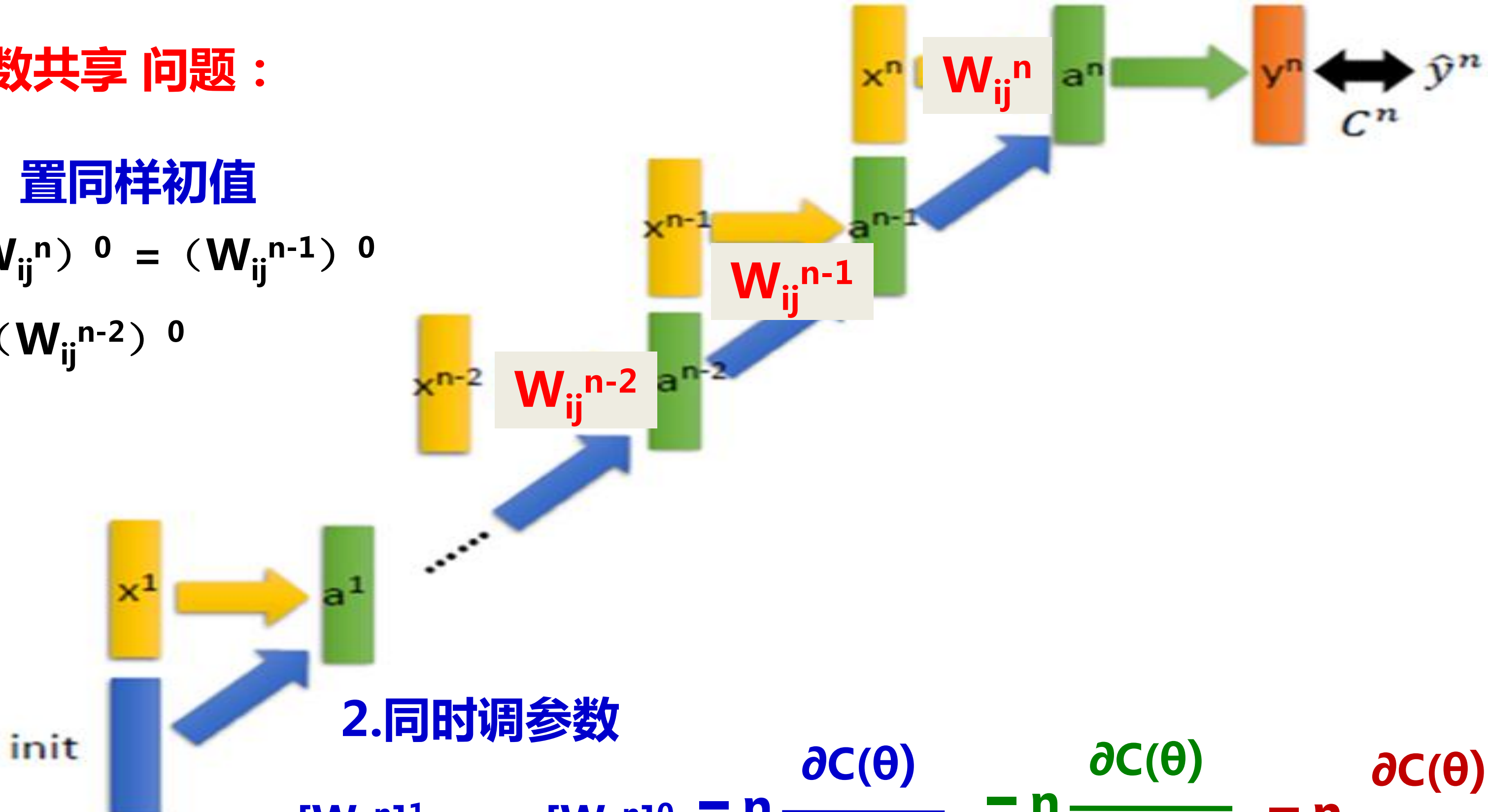
7.2 循环神经网络训练

参数共享 问题：

1. 置同样初值

$(W_{ij}^{n})^{0} = (W_{ij}^{n-1})^{0}$

$= (W_{ij}^{n-2})^{0}$



2.同时调参数

$$[W_{ij}^{n}]^{1} \leftarrow [W_{ij}^{n}]^{0} - \eta \frac{\partial C(\theta)}{\partial W_{ij}^{n}} - \eta \frac{\partial C(\theta)}{\partial W_{ij}^{n-1}} - \eta \frac{\partial C(\theta)}{\partial W_{ij}^{n-2}}$$

the same memory

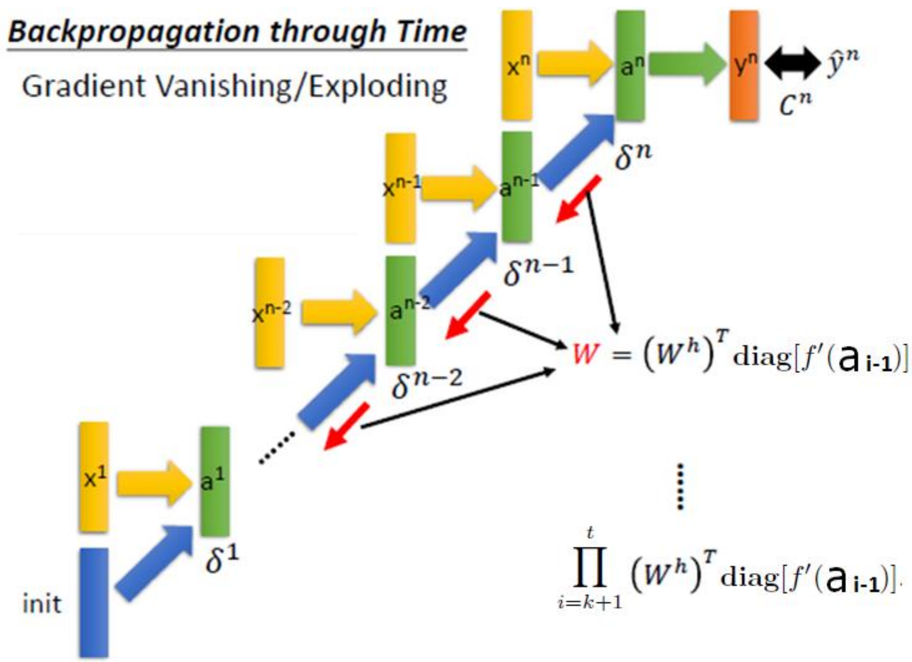
$$[W_{ij}^{n-1}]^{1} \leftarrow [W_{ij}^{n-1}]^{0} - \eta \frac{\partial C(\theta)}{\partial W_{ij}^{n-1}} - \eta \frac{\partial C(\theta)}{\partial W_{ij}^{n}} - \eta \frac{\partial C(\theta)}{\partial W_{ij}^{n-2}}$$

$$[W_{ij}^{n-2}]^{1} \leftarrow [W_{ij}^{n-2}]^{0} - \eta \frac{\partial C(\theta)}{\partial W_{ij}^{n-2}} - \eta \frac{\partial C(\theta)}{\partial W_{ij}^{n}} - \eta \frac{\partial C(\theta)}{\partial W_{ij}^{n-1}}$$

$\vdots$

7.2　循环神经网络训练

梯度消失/爆炸 问题

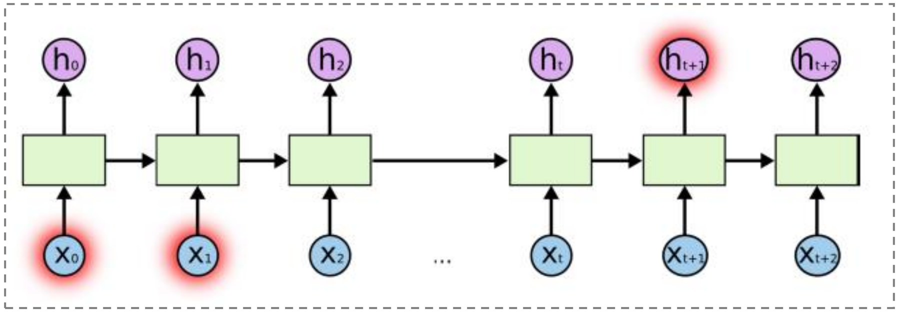


$\prod_{i=k+1}^{t} (W^h)^T \mathrm{diag}[f'(a_{i-1})]$ >1 梯度爆炸问题；相反，如果 <1，会出现和深度前馈神经网络类似的梯度消失问题。

在训练循环神经网络时，更经常出现的是梯度消失问题，训练较难

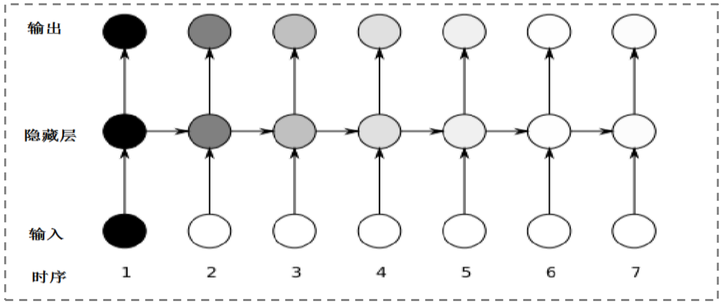
循环神经网络的长期依赖问题

问题：距当前节点越远的节点对当前节点处理影响越小，无法建模长时间的依赖



例如：

- The cat, which already ate a bunch of food, (was) full.
- The cats, which already ate a bunch of food, (were) full.



解决方法： → **LSTM、GRU 等**

内 容 提 要

7.0 概述

7.1 循环神经网络结构

7.2 循环神经网络训练

7.3 循环神经网络改进及变形

7.4 Encoder-Decoder 框架 RNN

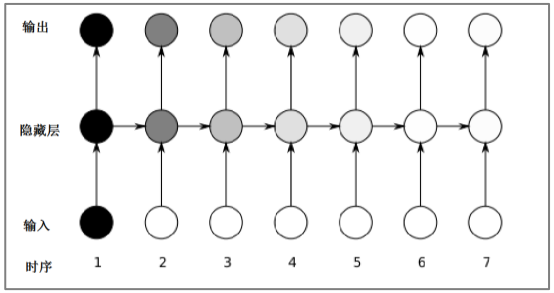
7.5 循环神经网络应用

7.3 循环神经网络改进及变形（内部单元）

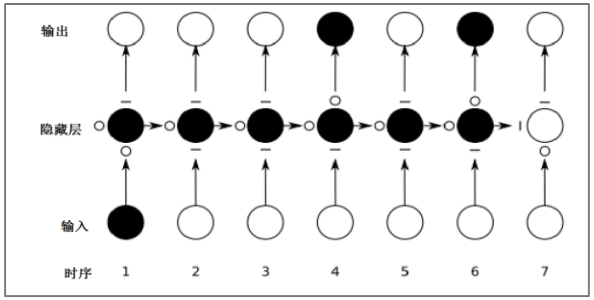
1. 长短时记忆神经网络：LSTM

LSTM基本思想

LSTM单元不仅接受 x_t 和 h_{t-1}，还需建立一个机制能保留前面远处结点信息不会被丢失



RNN

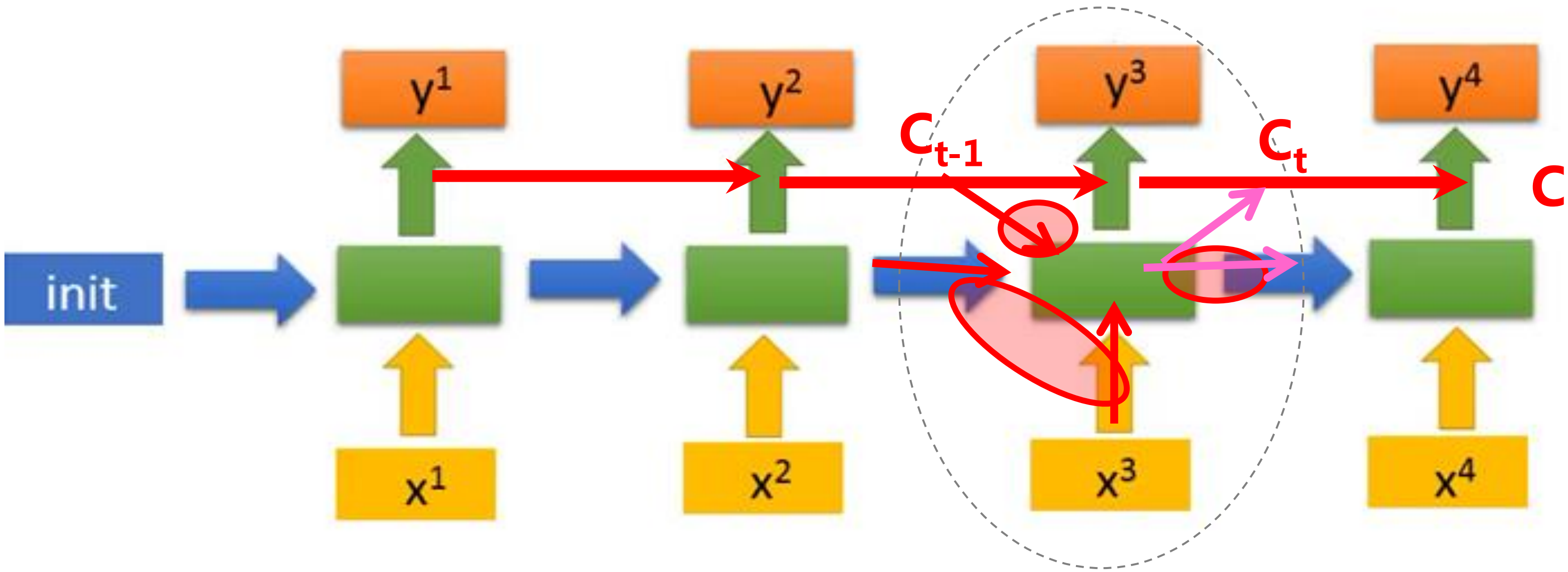


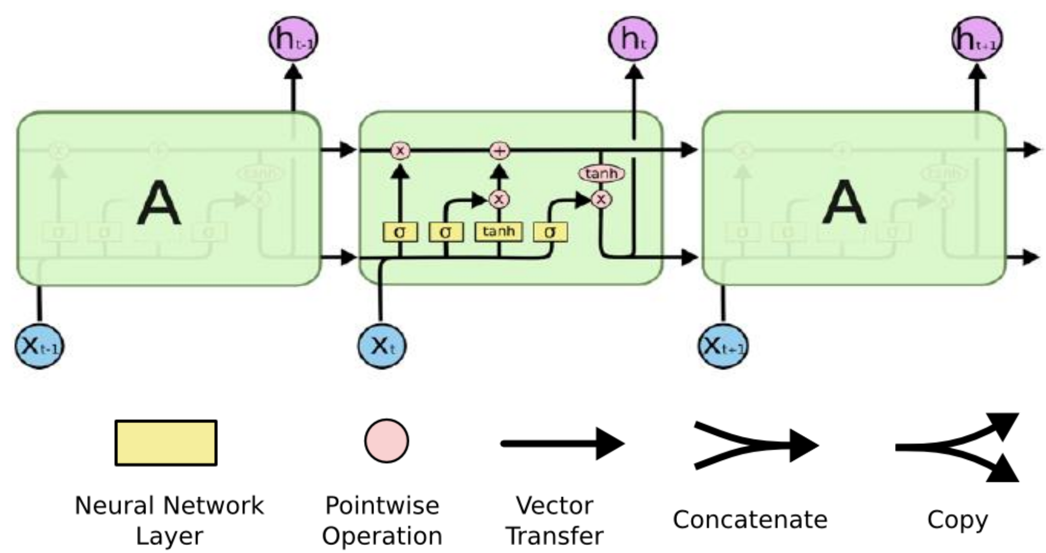
LSTM

LSTM 通过设计“门”结构实现保留信息和选择信息功能，每个门结构由一个sigmoid 层和一个poinewise操作构成

7.3 循环神经网络改进及变形（内部单元）

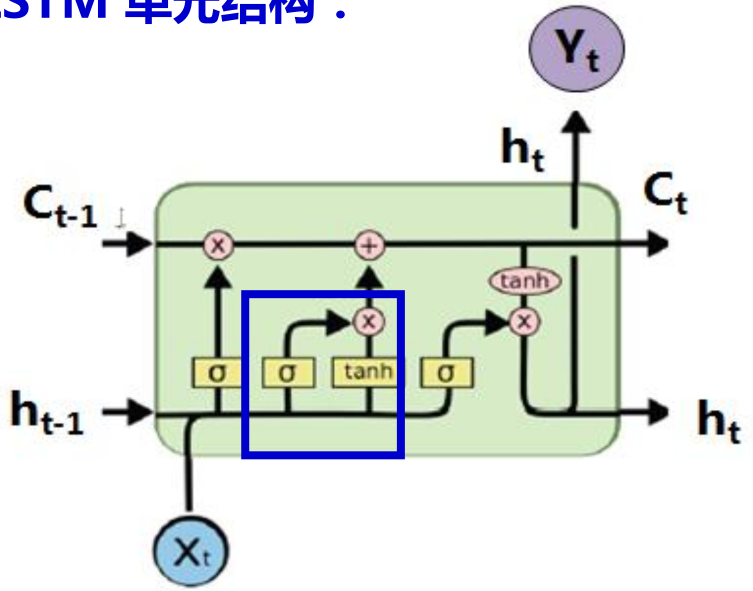
LSTM





7.3 循环神经网络改进及变形（内部单元）

LSTM 单元结构：



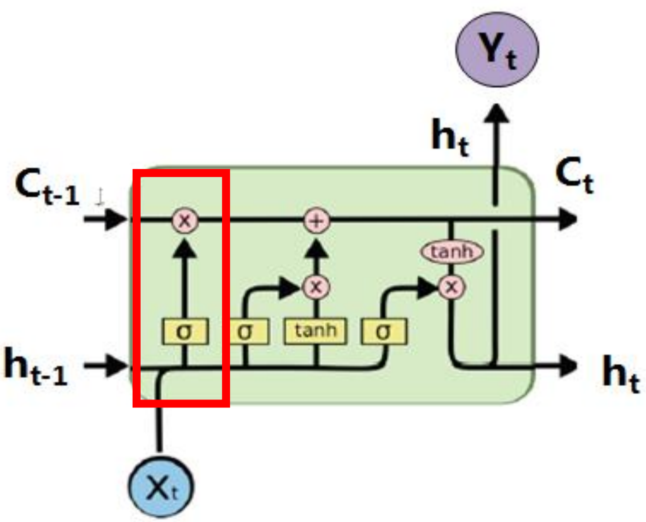
输入门 i_t

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

（决定加入多少新信息）

输入产生新信息:

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



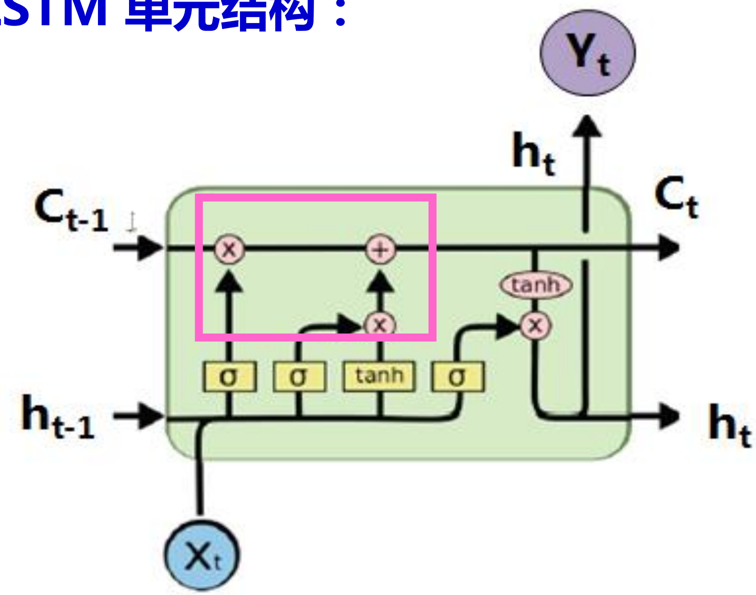
遗忘门 f_t：

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

(决定丢弃多少旧信息)

7.3 循环神经网络改进及变形（内部单元）

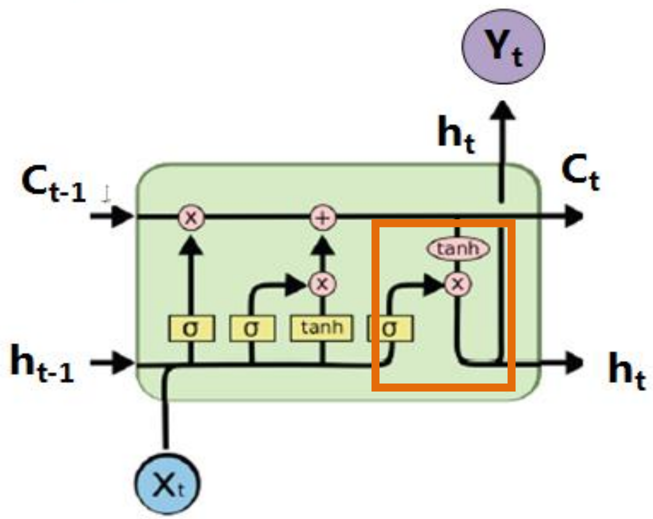
LSTM 单元结构：



当前保留信息

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

(在生成的**当前保留信息**中输入产生 新信息和旧信息各占多少)



输出门 o_t

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

隐状态输出

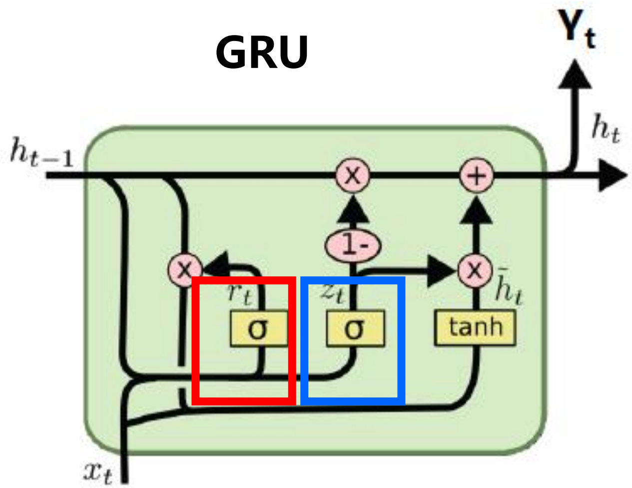
$$h_t = o_t * \tanh\left(C_t\right)$$

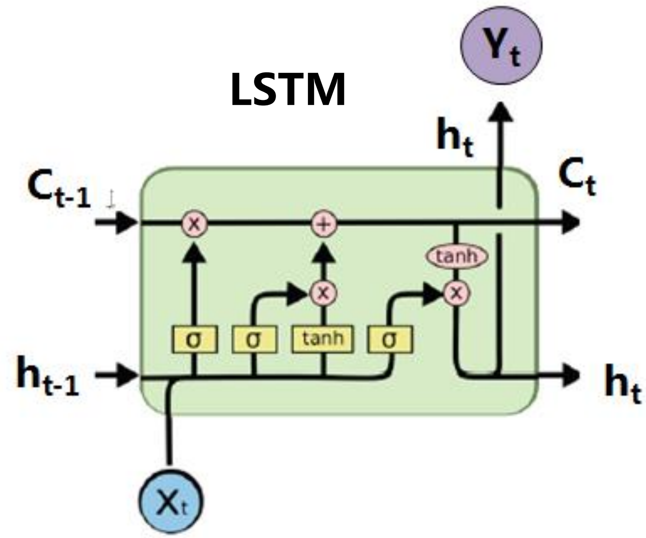
参数： W_f, W_i, W_o, W_C

7.3 循环神经网络改进及变形（内部单元）

2. LSTM 简化 GRU

输入门和遗忘门合并为更新门（更新门决定隐状态保留放弃部分）





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

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

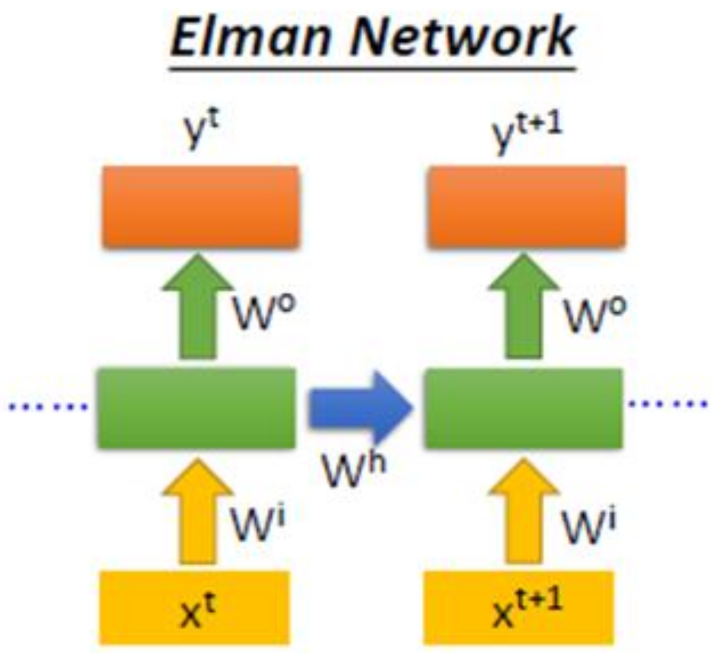
新信息： $\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$

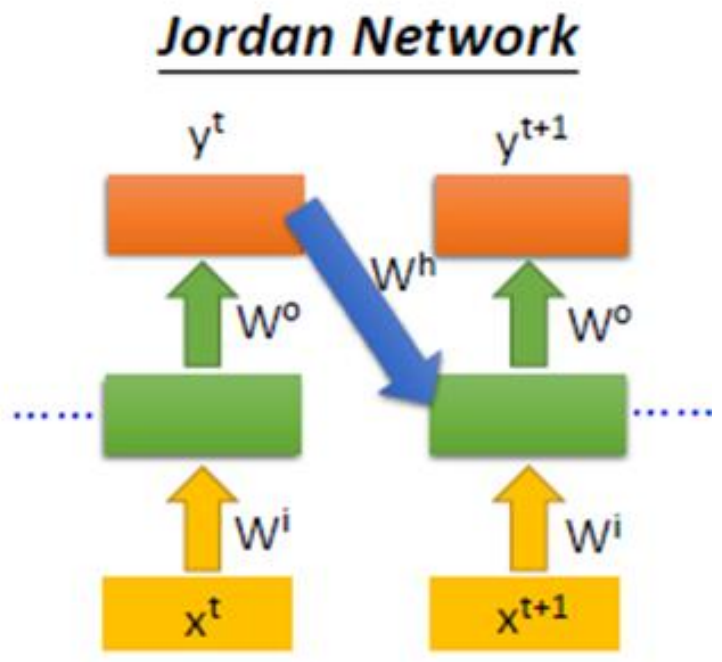
隐状态： $h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$

参数：

W_z ,W_r ,W

7.3　循环神经网络改进及变形（网络结构）





$h(t) = \sigma(W_i X + W_h\ h(t-1) + b\ )$

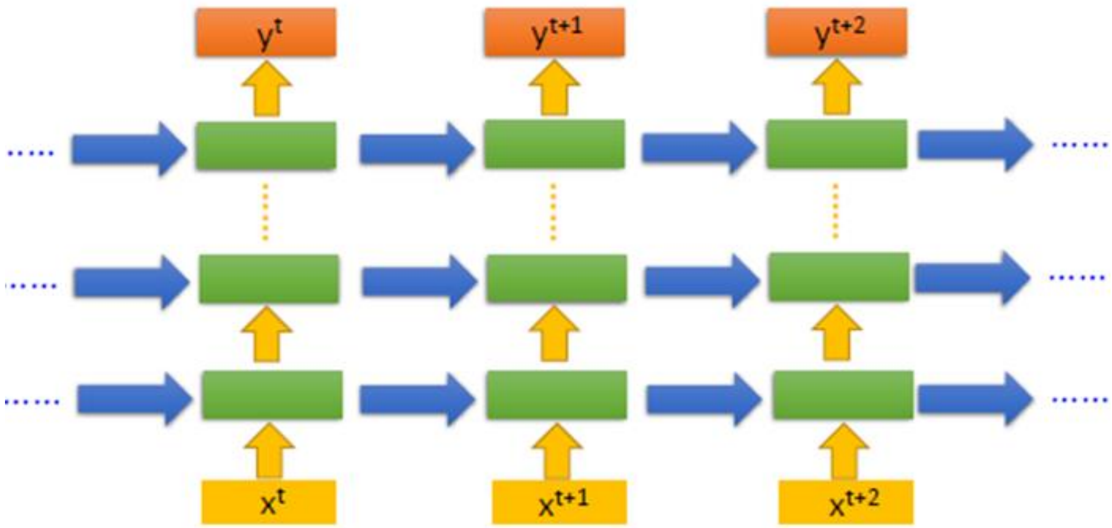
$Y = \mathrm{softmax}(W_o h(t))$

$h(t) = \sigma(W_i X + W_h\ Y(t-1) + b\ )$

$Y = \mathrm{softmax}(W_o h(t))$

7.3 循环神经网络改进及变形（网络结构）

. Deep RNN



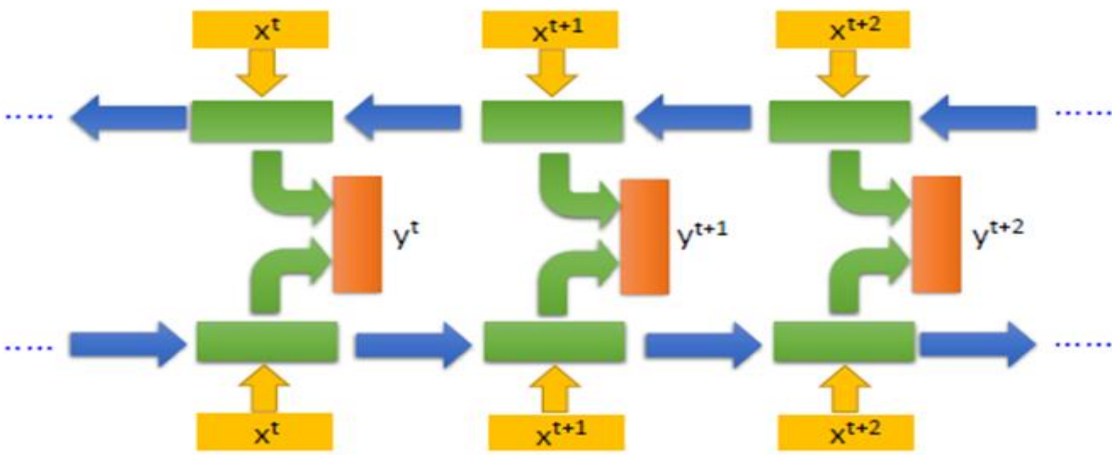
$$h^i(t) = \sigma(W^i{}_i\, h^{i-1}(t) + W^i{}_h\, h^i(t-1) + b^i)$$

$$Y = \mathrm{softmax}(W_o h^L(t))$$

深度RNN采用多个隐层，每个隐层向后一层传递序列信息

7.3 循环神经网络改进及变形（网络结构）

Bidirectional RNNs



$$\vec{h}_t = f(\overrightarrow{W}x_t + \vec{V}\vec{h}_{t-1} + \vec{b})$$

$$\overleftarrow{h}_t = f(\overleftarrow{W}x_t + \overleftarrow{V}\overleftarrow{h}_{t+1} + \overleftarrow{b})$$

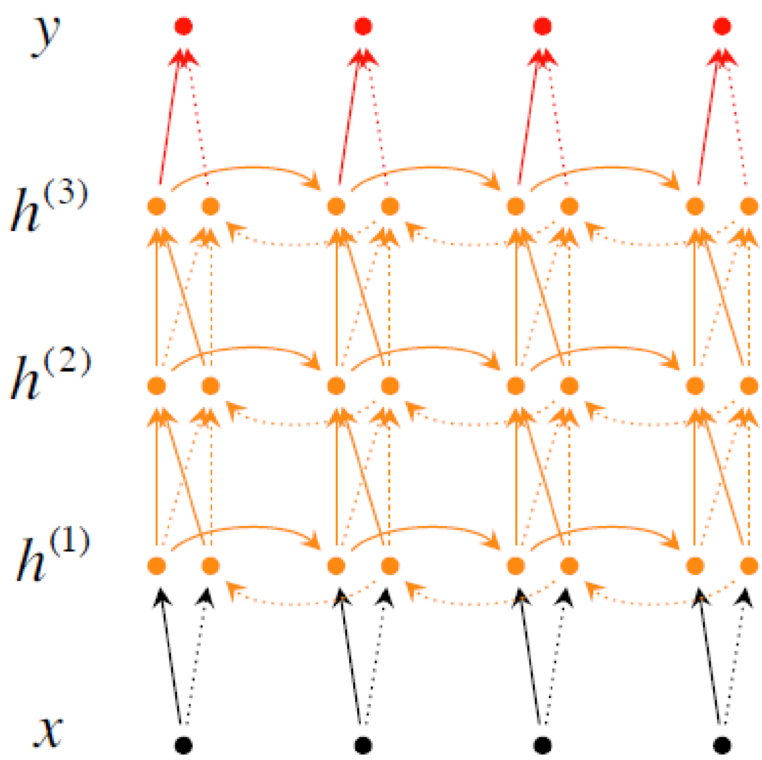
$$y_t = g(U[\vec{h}_t;\overleftarrow{h}_t] + c)$$

每个时刻都有一个正向输入的隐层 $\overrightarrow{ht}$ 和·一个反向输入隐层 $\overleftarrow{h_t}$

两个隐层分别可以表示一个词的上文信息和下文信息

7.3 循环神经网络改进及变形（网络结构）

. Deep Bidirectional RNN



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

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

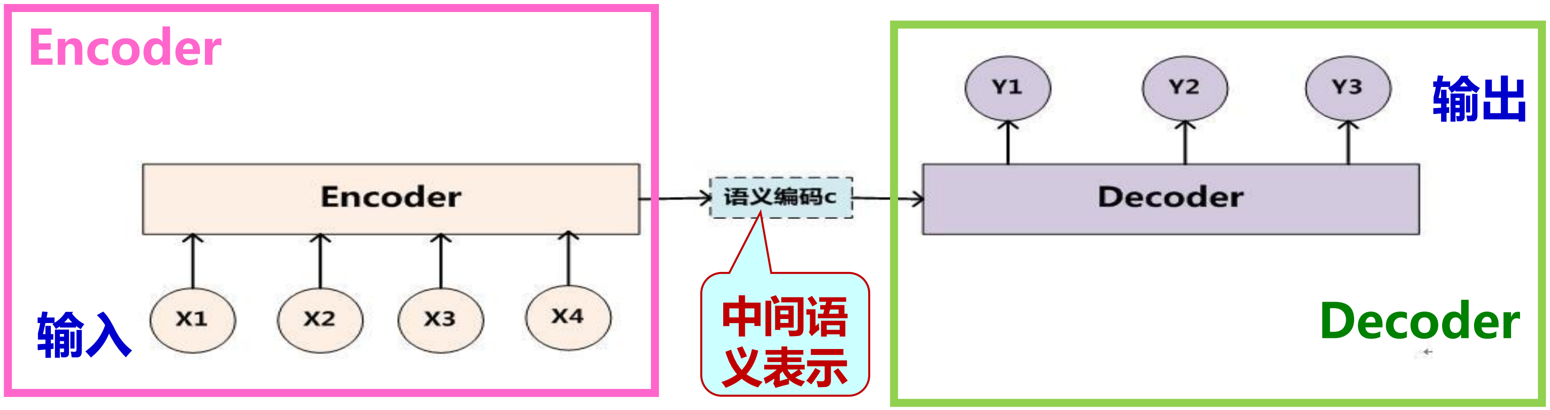
$$y_t = g(U[\overrightarrow{h}_t^{(L)}; \overleftarrow{h}_t^{(L)}] + c)$$

内 容 提 要

7.4 Encoder-Decoder 框架 RNN

Encoder-Decoder 框架

Encoder-Decoder是个非常通用的计算框架，抽象的表示：



Encoder：对输入X序列进行编码，通过非线性变换转化为

中间语义表示： $C = \mathcal{F}(x_1, x_2 \dots x_m)$

Decoder：根据X的中间语义表示C和已经生成的$y_1, y_2 \dots y_{i-1}$来生成

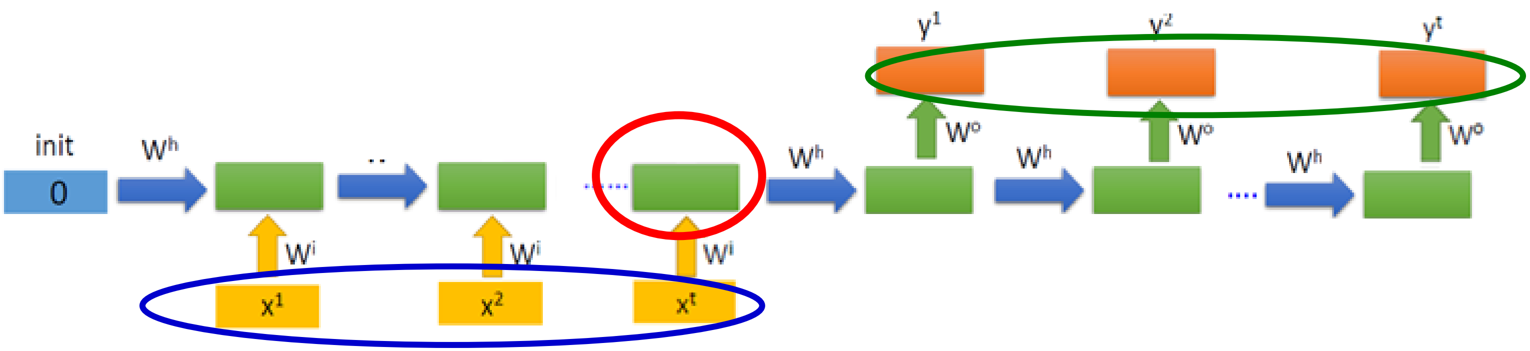
i 时刻的 y_i $y_i = \mathcal{G}(C, y_1, y_2 \dots y_{i-1})$

Encoder和Decoder具体使用什么模型都是由研究者自己确定，

如，CNN/RNN/BiRNN/GRU/LSTM/Deep LSTM等

17.5 Encoder-Decoder 框架 RNN

Encoder-Decoder 框架 RNN



输入 $X = \langle x_1, x_2 \dots x_m \rangle$

输出 $Y = \langle y_1, y_2 \dots y_n \rangle$

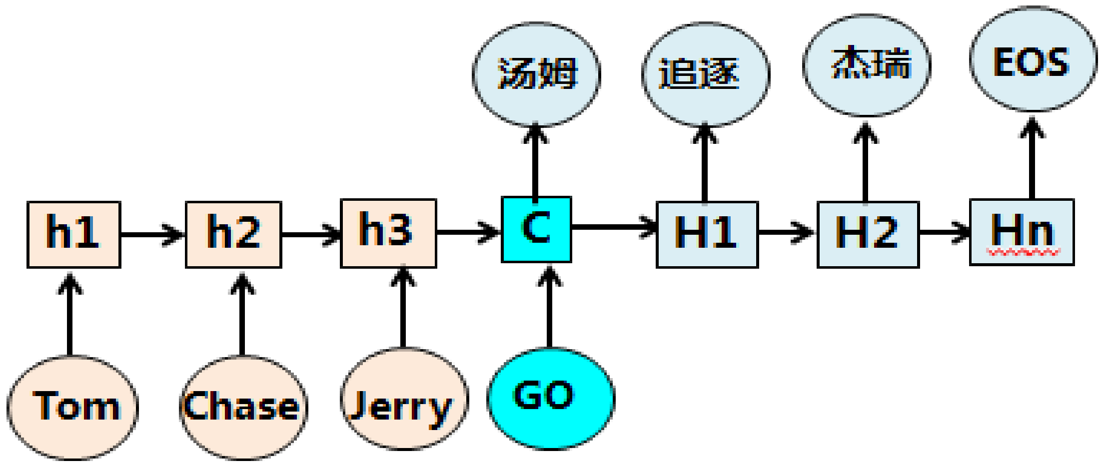
中间语义表示 $C = \mathcal{F}(x_1, x_2 \dots x_m)$

$$y_1 = f(C)$$
$$y_2 = f(C, y_1)$$
$$y_3 = f(C, y_1, y_2)$$
$$y_i = \mathcal{G}(C, y_1, y_2 \dots y_{i-1})$$

C 可以作为X的句向量

17.5 Encoder-Decoder 框架 RNN

例:



$$X = \langle x_1, x_2 \dots x_m \rangle$$
$$Y = \langle y_1, y_2 \dots y_n \rangle$$

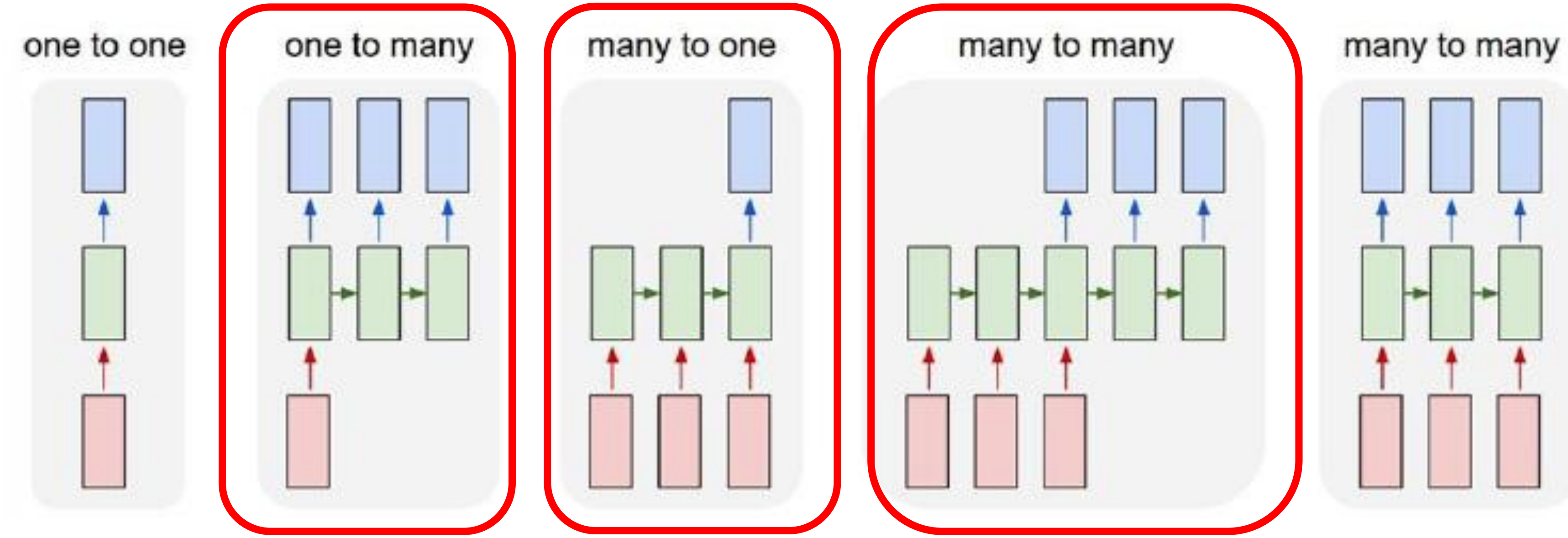
$$C = \mathcal{F}(x_1, x_2 \dots x_m)$$

$$y_1 = f(C)$$
$$y_2 = f(C, y_1)$$
$$y_3 = f(C, y_1, y_2)$$
$$y_i = \mathcal{G}(C, y_1, y_2 \dots y_{i-1})$$

Encoder-Decoder 框架RNN

17.5 Encoder-Decoder 框架 RNN

RNN 输入输出结构：

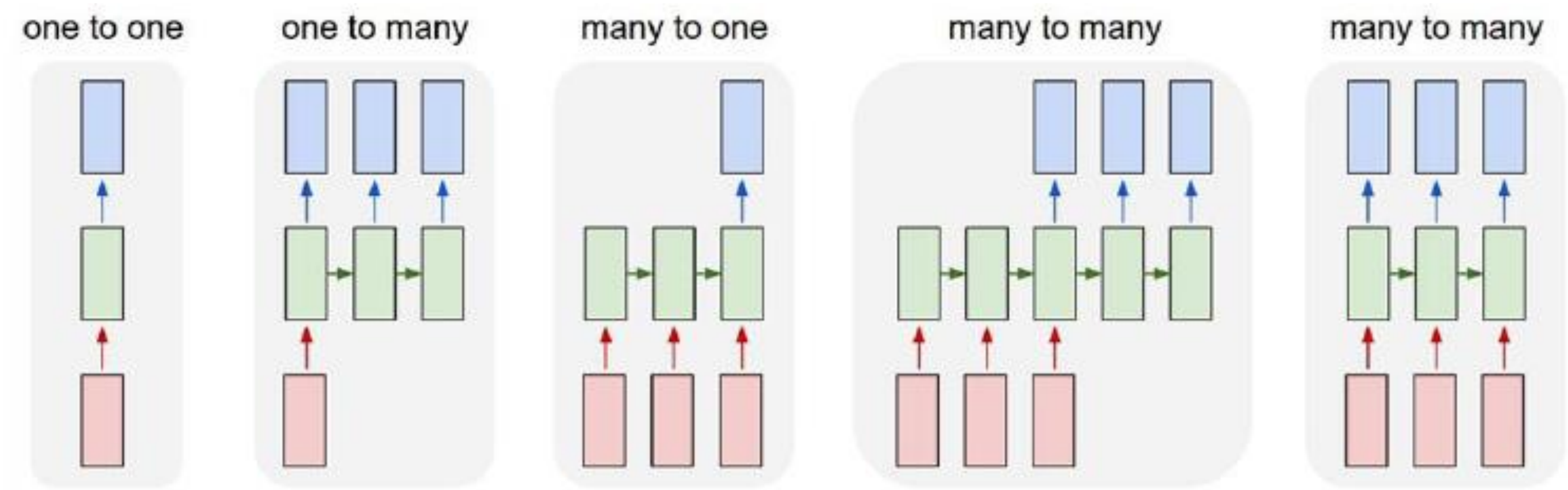


一般Encoder和Decoder都采用RNN模型，对于句子比较长的情形，LSTM和GRU模型效果要明显优于RNN模型。但当句子长度超过30以后，LSTM模型的效果会急剧下降，一般此时会引入Attention模型。

内 容 提 要

17.6 循环神经网络应用

RNN/LSTM 建模的序列问题



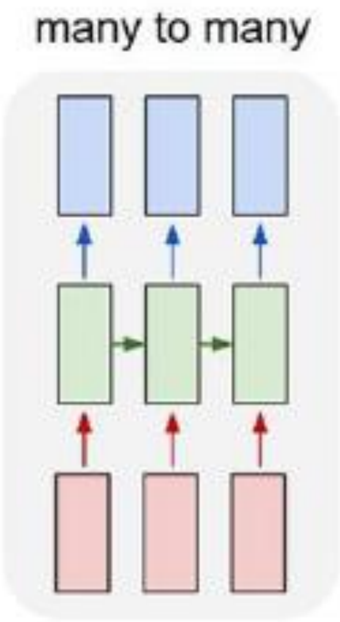
图像分类　图文标注　情感分类　机器翻译　序列标注

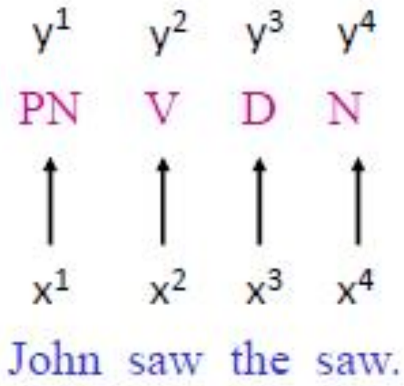
17.6 循环神经网络应用

如：

POS Tagging

- Input and output are vector sequences with ***the same length***

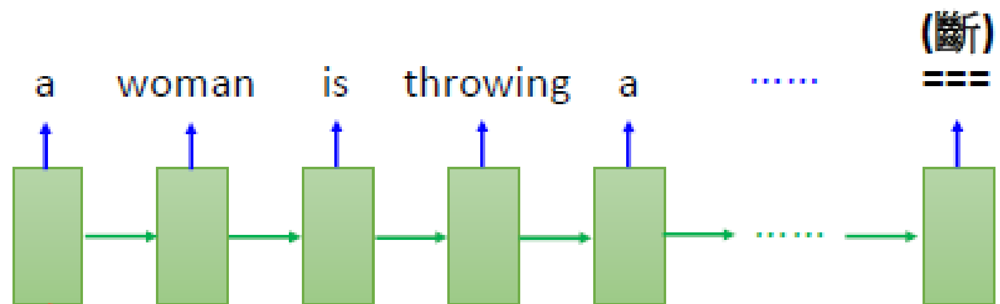




17.6 循环神经网络应用

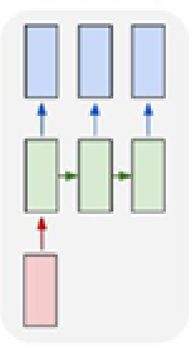
Caption generation

- Input is one vector, but output is a vector sequence



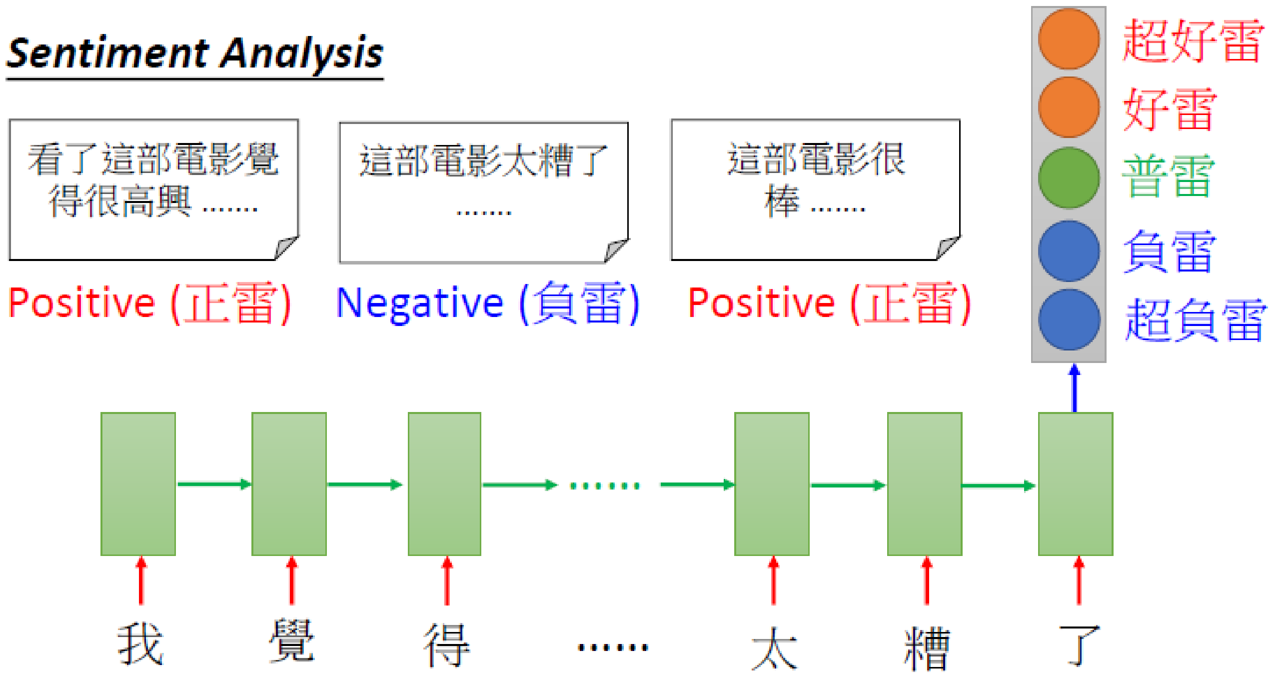
Input

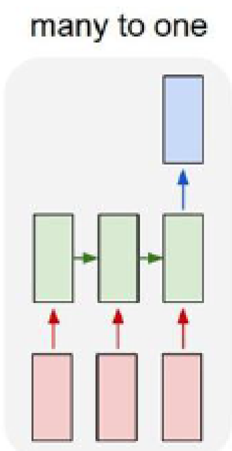
one to many



Many to one

- Input is a vector sequence, but output is only one vector



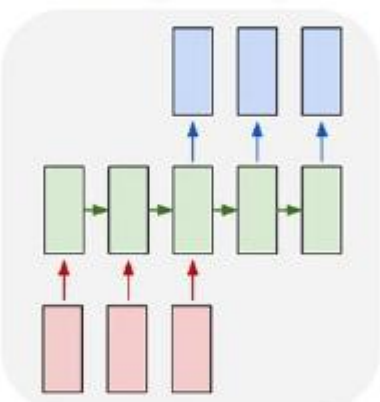


情感分析

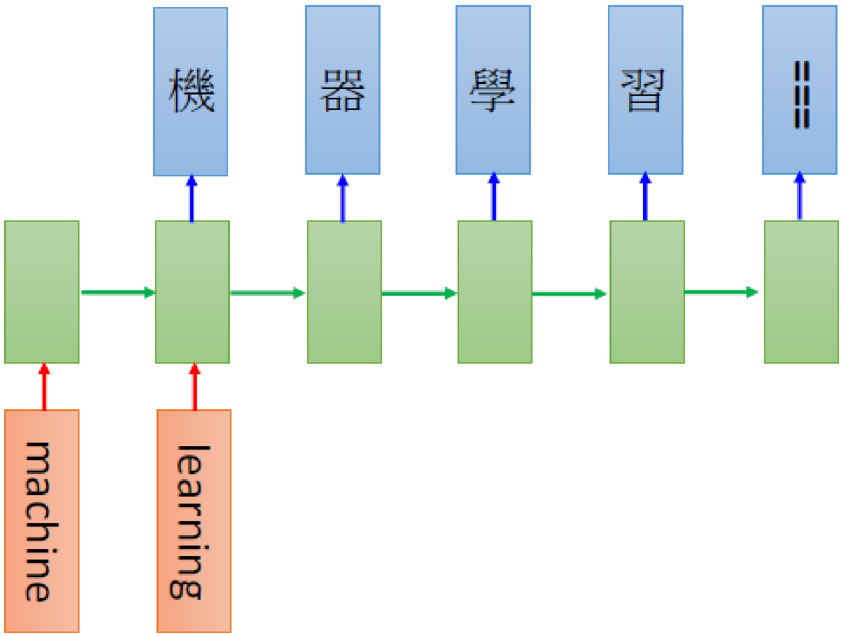
17.6 循环神经网络应用

- Both input and output are vector sequences ***with different lengths***. → ***Sequence to sequence learning***

many to many



机器翻译



参考文献：

李宏毅课程

http://speech.ee.ntu.edu.tw/~tlkagk/courses_ML16.html

邱锡鹏，《神经网络与深度学习》讲义

刘鹏飞 ，卷积神经网络和递归 神经网络实践

刘昕 ，深度学习一线实战暑期研讨班 深度学习基础

在 此 表 示 感 谢！

谢谢各位！