循环神经网络RNN & LSTM

——经典NLP模型

SJTU

目录 / CONTENTS

01

循环神经网络 Recurrent Neural Network

学习目标:

- **掌握**RNN网络结构与相关概念
- **理解**循环神经网络的优劣势与应用场景

Recurrent Neural Network

有这样一类应用场景，如：智能对话机器人的简易开发

管家，帮我订一张下午3:00的电影票

如何响应和执行？

我是一个没有感情的机器人
需要为你做点什么吗

Slot Filling
填槽

- number：一张
- time：3:00
- task：电影票

在开发的时候

设置了哪些Slot

来获取信息

这就是开发者要做的事情
——判断每一个词属于哪个slot
(分类问题)

怎么把单词喂给网络?

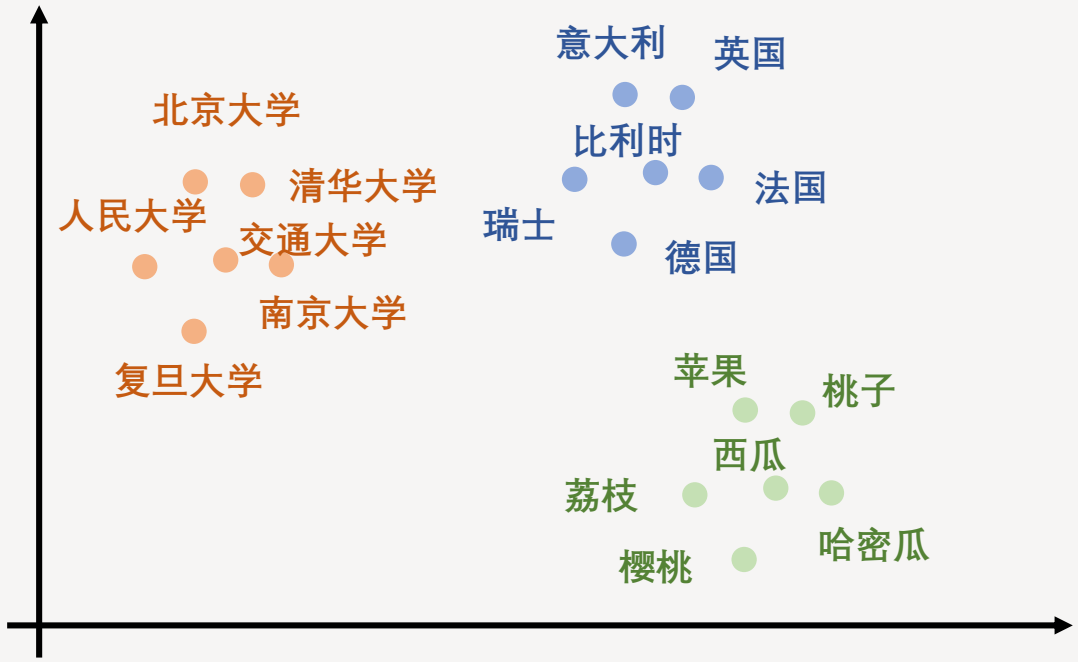
One-hot Vector（1 of N encoding)

V = {wish, have, a, great, day}.

wish = [1,0,0,0,0]
have = [0,1,0,0,0]
a = [0,0,1,0,0]
great = [0,0,0,1,0]
day = [0,0,0,0,1]

Word Embedding

我要一杯橘子味的汽水
我要一杯橙子味的汽水



语义上相近的单词在新的空间上距离很近

缺点

- 向量的维度太大，浪费资源
- 任意两个词之间是孤立的，无法表示语义层面上词汇间的相关信息（key point)

回到刚刚slot fitting的话题

假如你需要订一张机票…

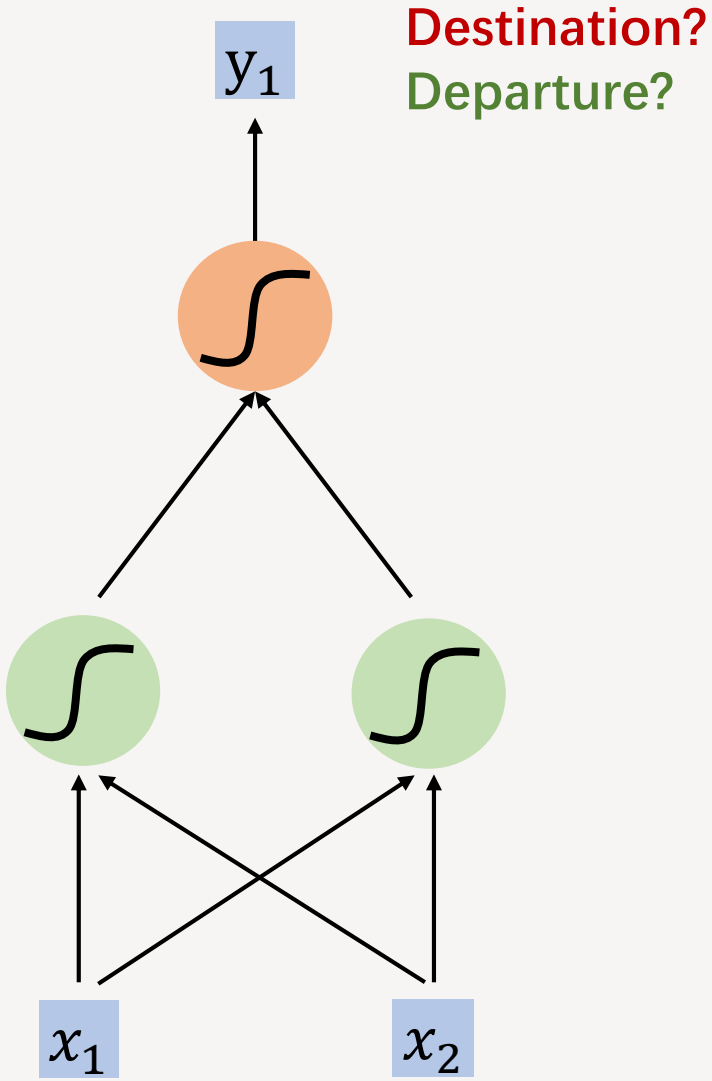
arrive Shanghai on May 20th

destination time time

leave Shanghai on May 20th

departure time time

这时候你的神经网络
需要有 记 忆 性!



RNN就是这样一个有“记忆性”的网络

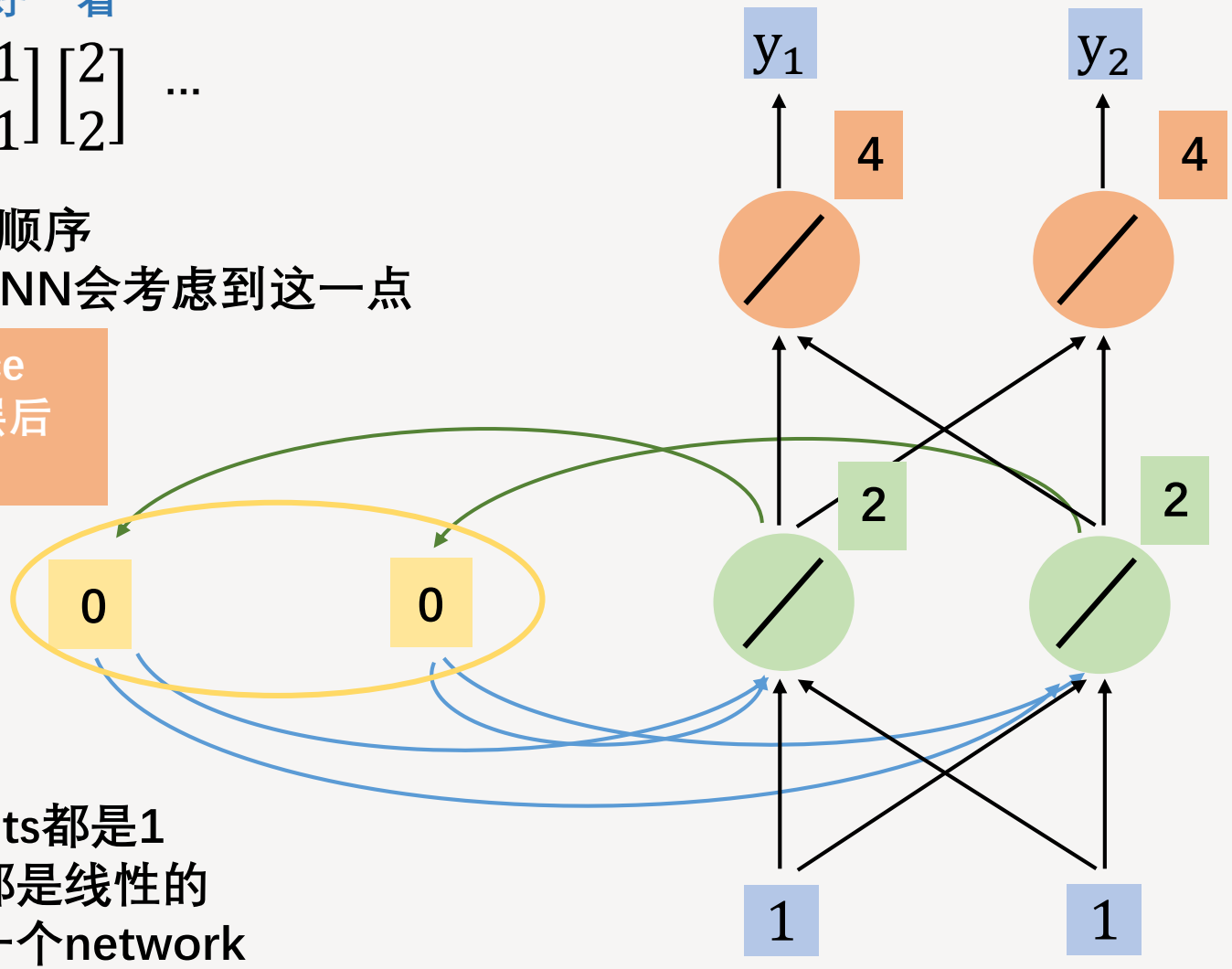
好 好 看

有这样一串输入：$\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}\cdots$

传统的NN训练样本没有顺序
语言输入是有顺序的，RNN会考虑到这一点

开辟一个Memory Space
把上一个节点经过隐藏层后
计算得到的值存下来

给memory一组初始值



Simple RNN

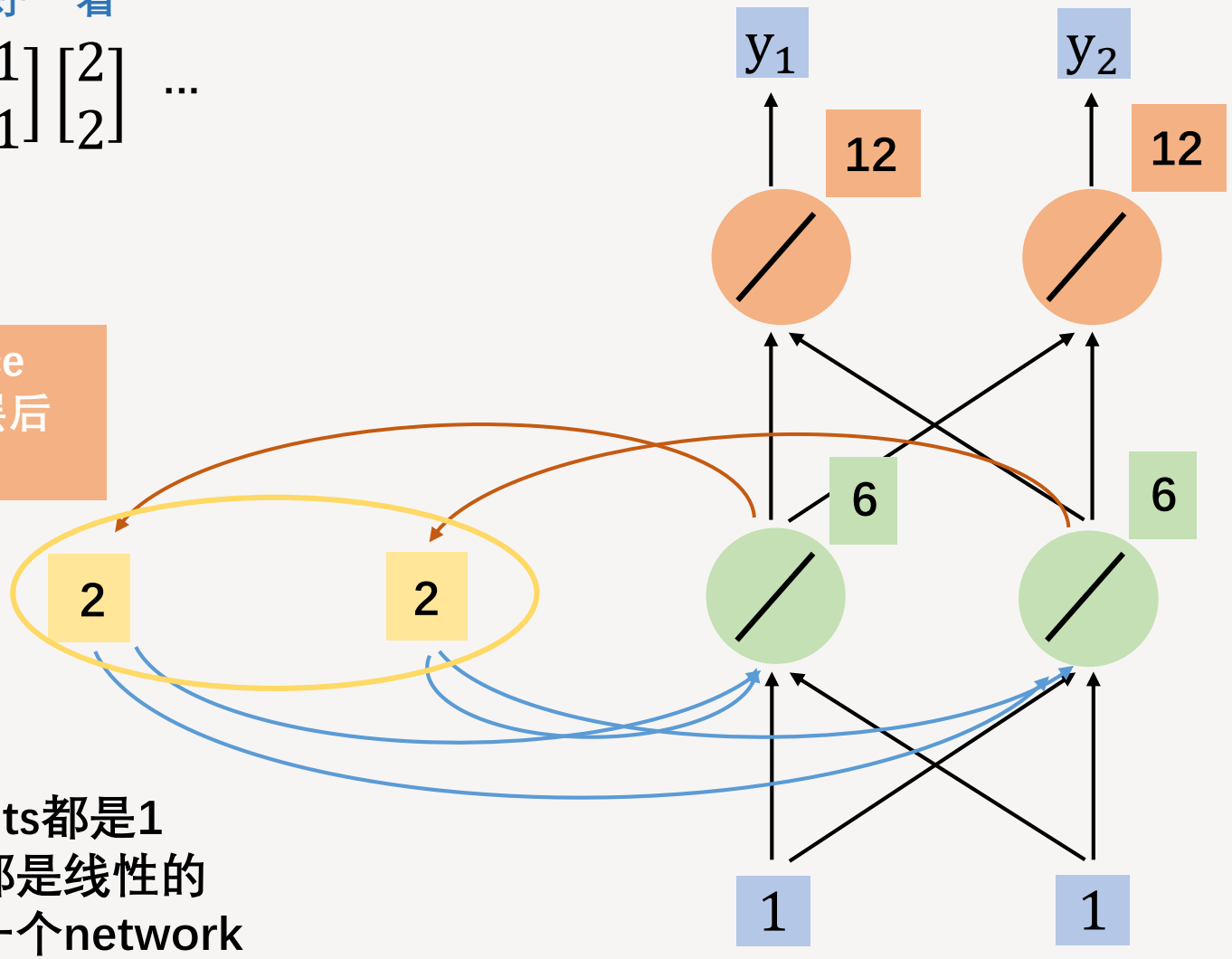
- 假设所有的权值weights都是1
- 假设所有的激活函数都是线性的
- 在不同时间点使用同一个network

RNN就是这样一个有“记忆性”的网络

好 好 看

有这样一串输入：$\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}\cdots$

开辟一个Memory Space
把上一个节点经过隐藏层后
计算得到的值存下来



- 假设所有的权值weights都是1
- 假设所有的激活函数都是线性的
- 在不同时间点使用同一个network

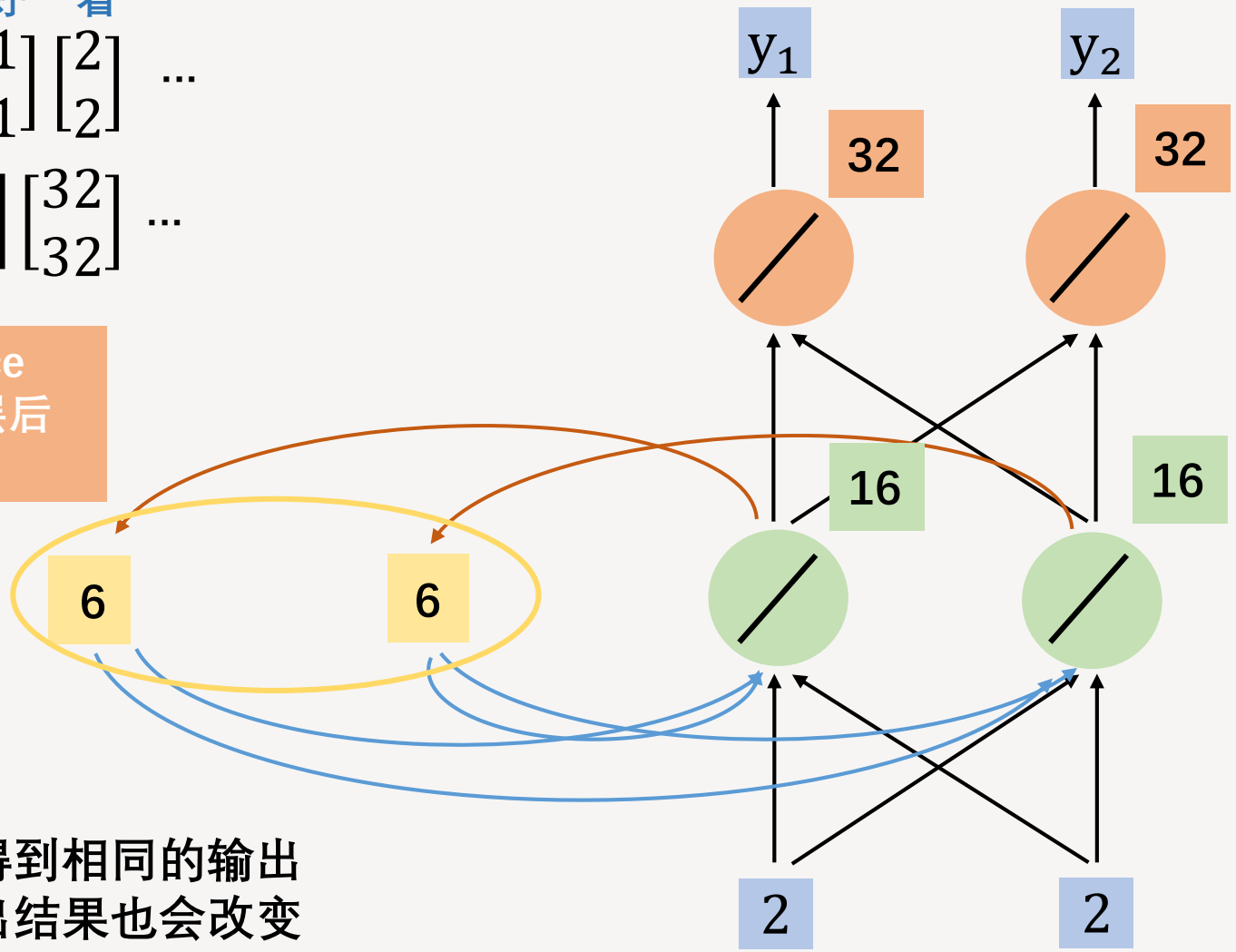
RNN就是这样一个有“记忆性”的网络

好 好 看

有这样一串输入：$\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}\ldots$

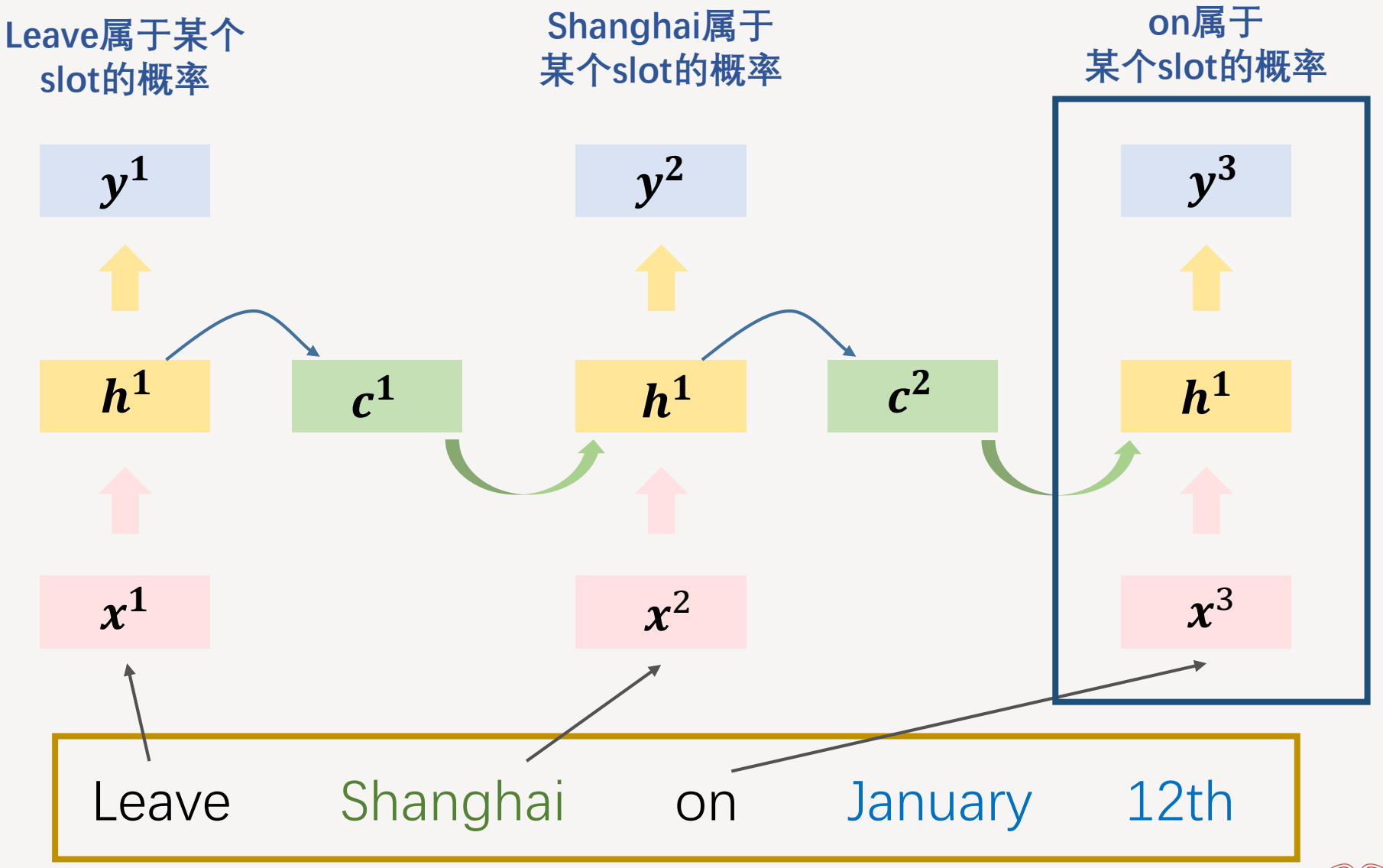
得到的输出为：$\begin{bmatrix}4\\4\end{bmatrix}\begin{bmatrix}12\\12\end{bmatrix}\begin{bmatrix}32\\32\end{bmatrix}\ldots$

开辟一个Memory Space
把上一个节点经过隐藏层后
计算得到的值存下来



- 相同的输入不一定会得到相同的输出
- 输入的顺序改变，输出结果也会改变

RNN的计算过程

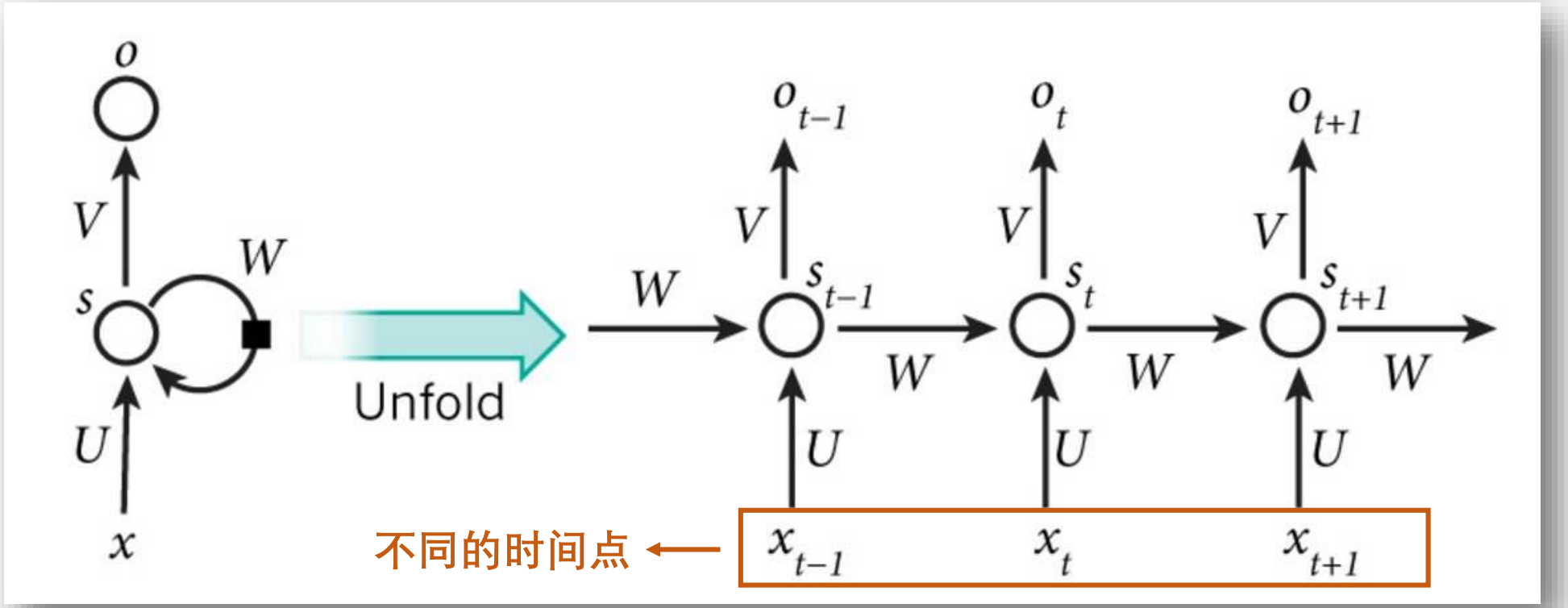


一组输入的每一个vector用同一个网络进行训练

Recurrent Neural Network

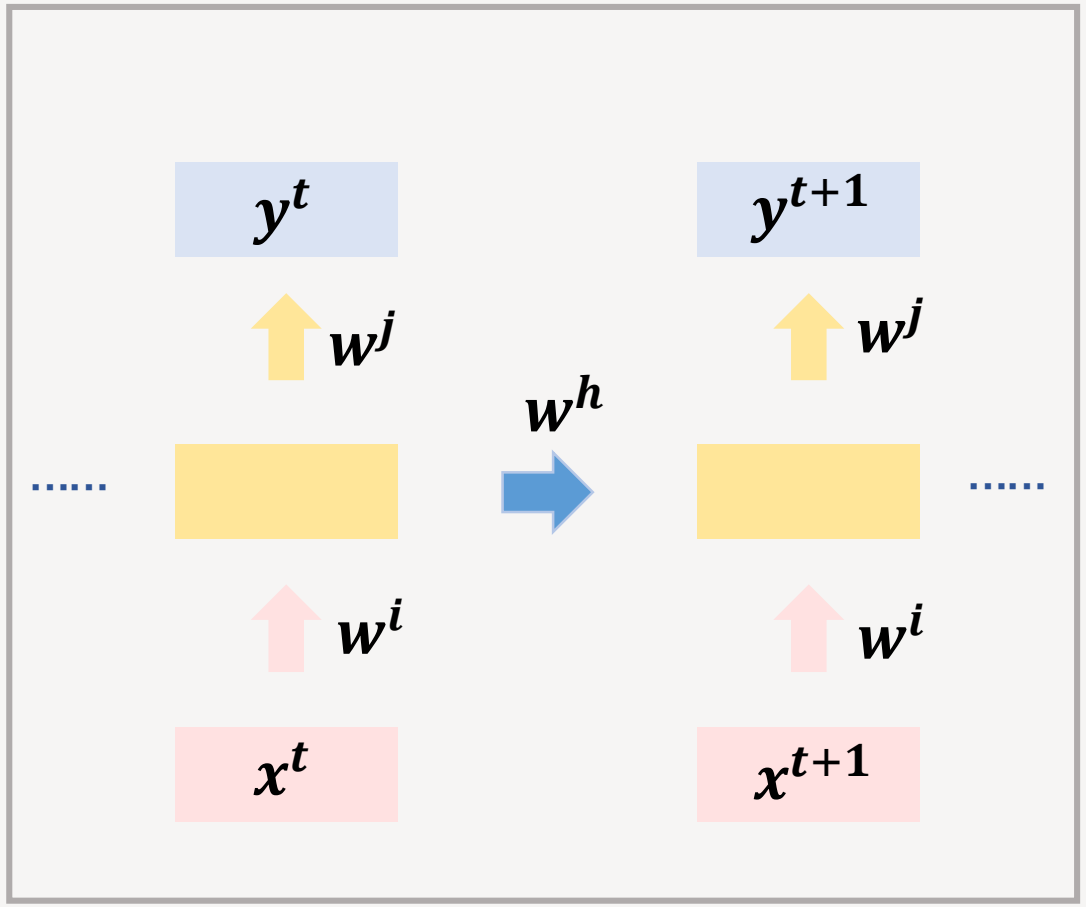
大家在网上看到的RNN可能长这个样子

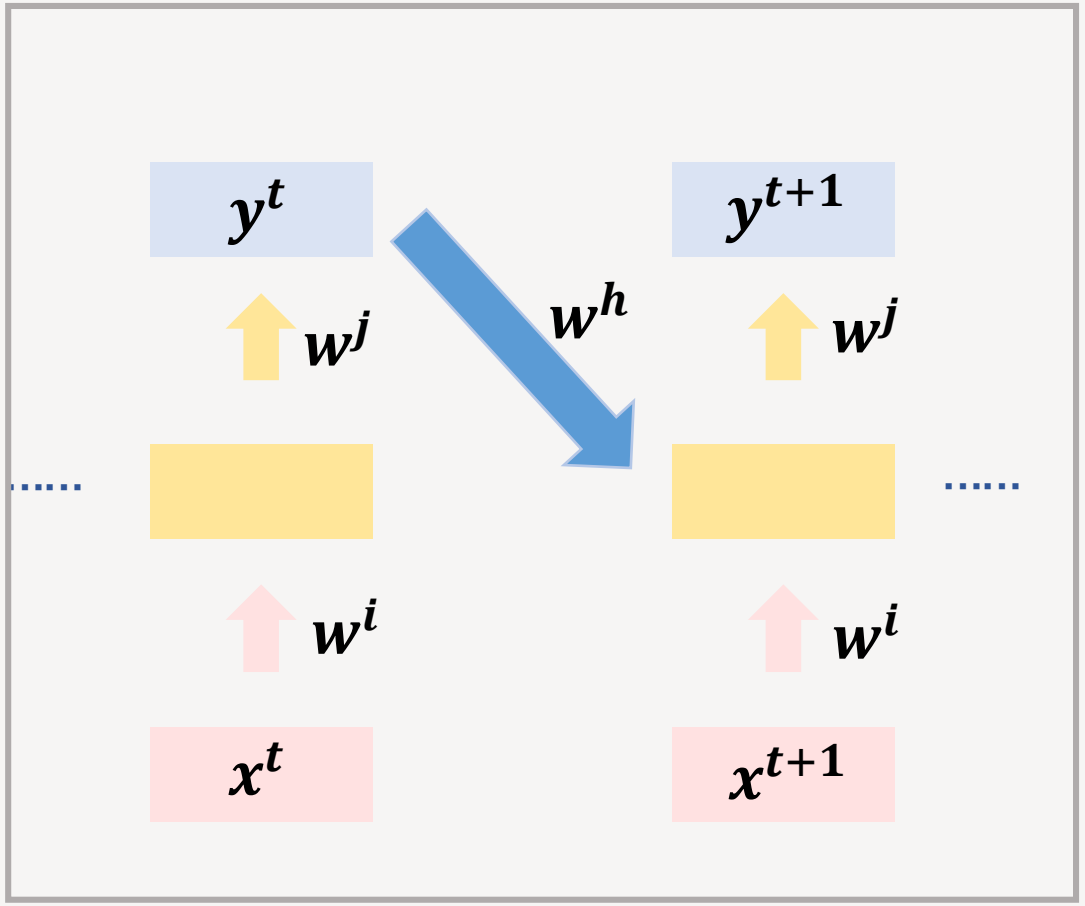
RNN也可以是deep的!



Memory存放哪种Hidden Layer的输出

理论来说，第二种学到的东西更准确



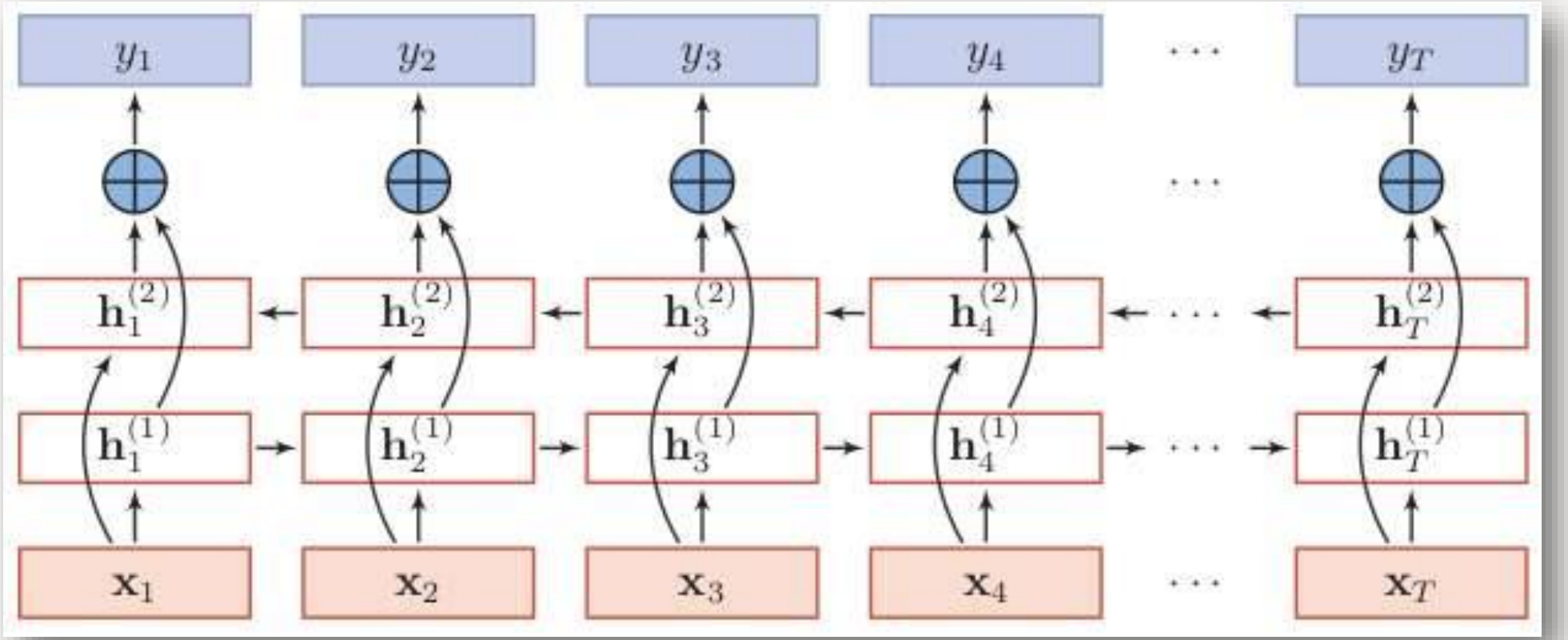


一个新的问题——怎么解决根据上下文的填空？比如，“今天我起的很 ___，为了看日出”

Bidirectional RNN

双向的循环神经网络

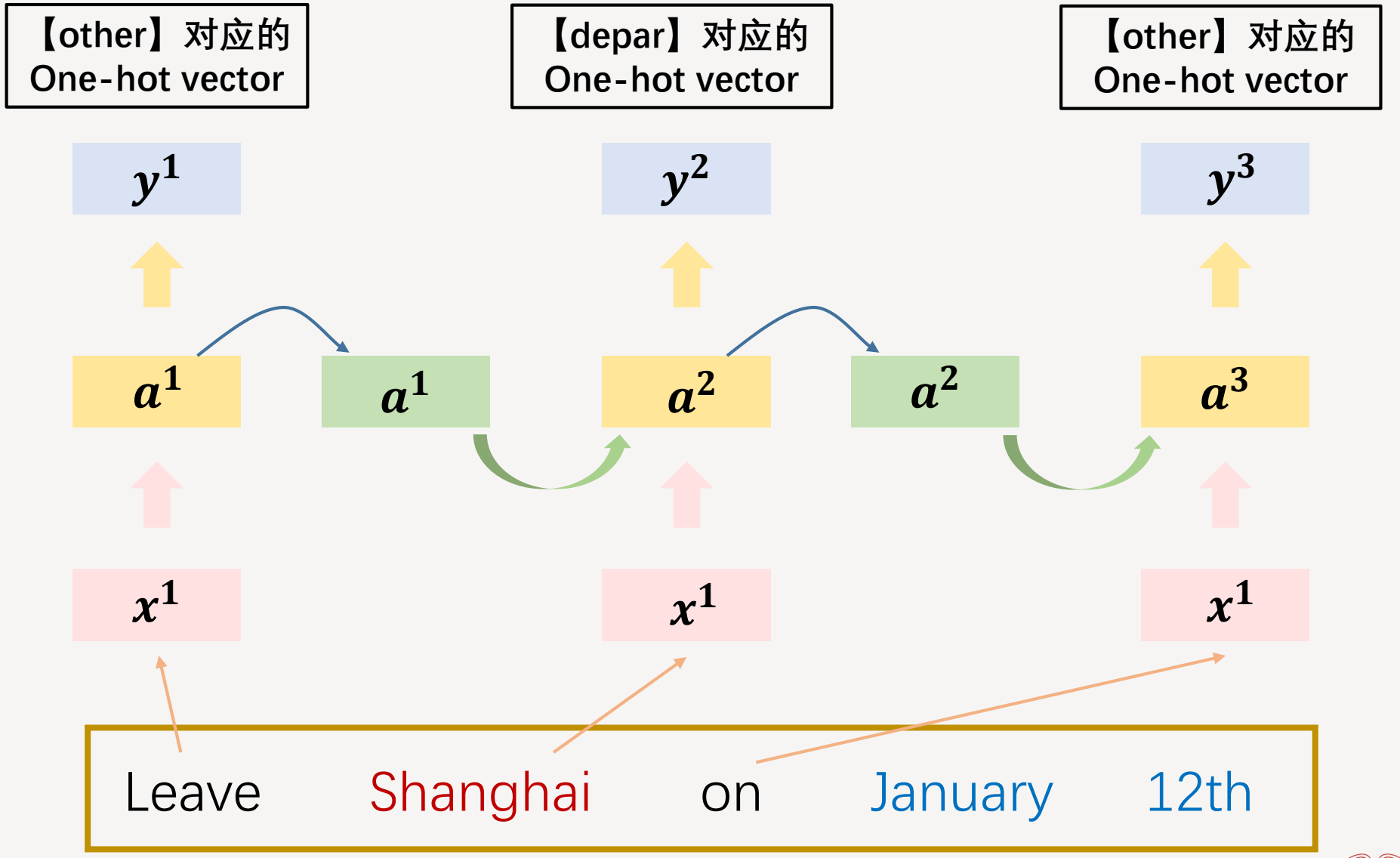
这一刻不仅看过前面的信息，还看过后面的信息，比只看一半效果更好



从前往后

从后往前

怎么训练 Recurrent Neural Network



BPTT
(Back Propagation Trough Time)

目录 / CONTENTS

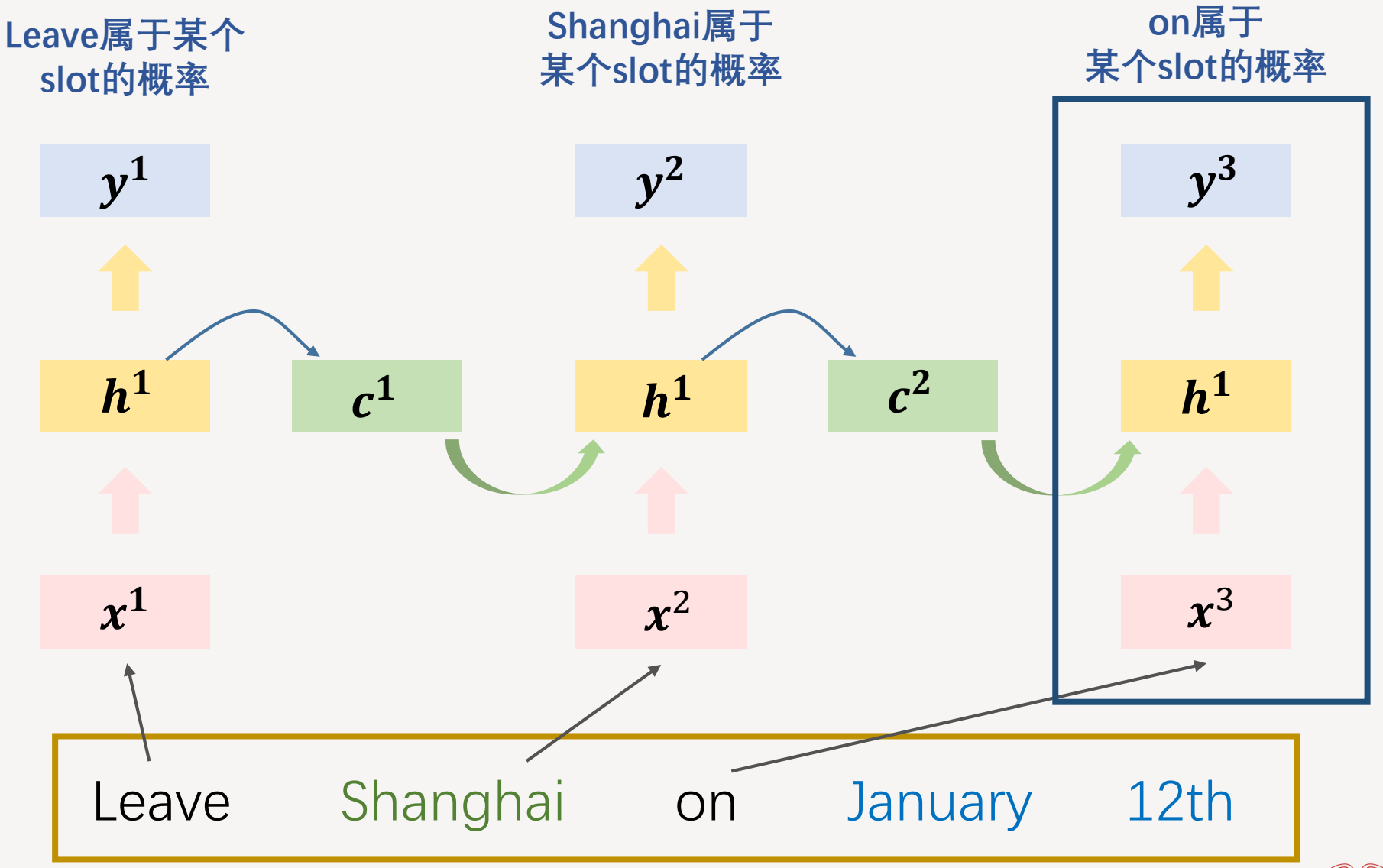
02

长短期记忆
LSTM

学习目标:

- **掌握**LSTM网络结构与相关概念
  （各类gate、线性变换等）
- **能够分析**LSTM与RNN比较的优缺点
- **理解**Sequence to sequence的应用场景

RNN的计算过程

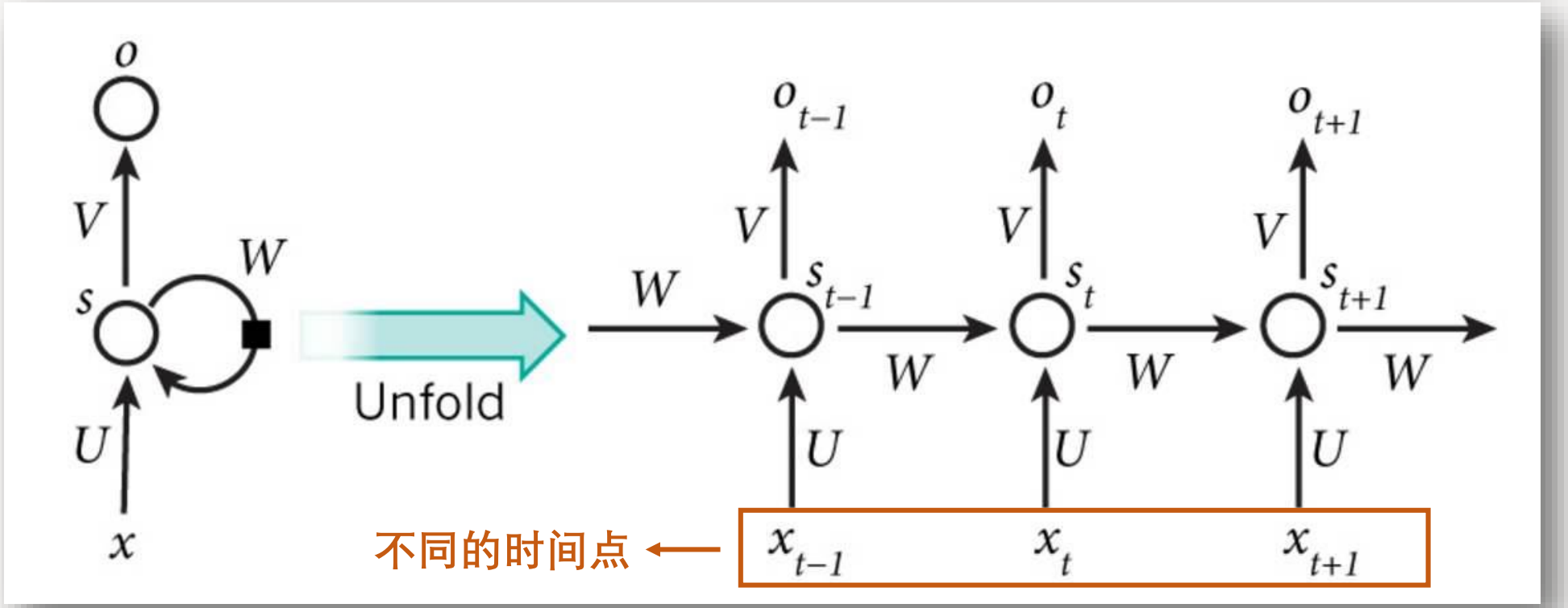


一组输入的每一个vector用同一个网络进行训练

Recurrent Neural Network

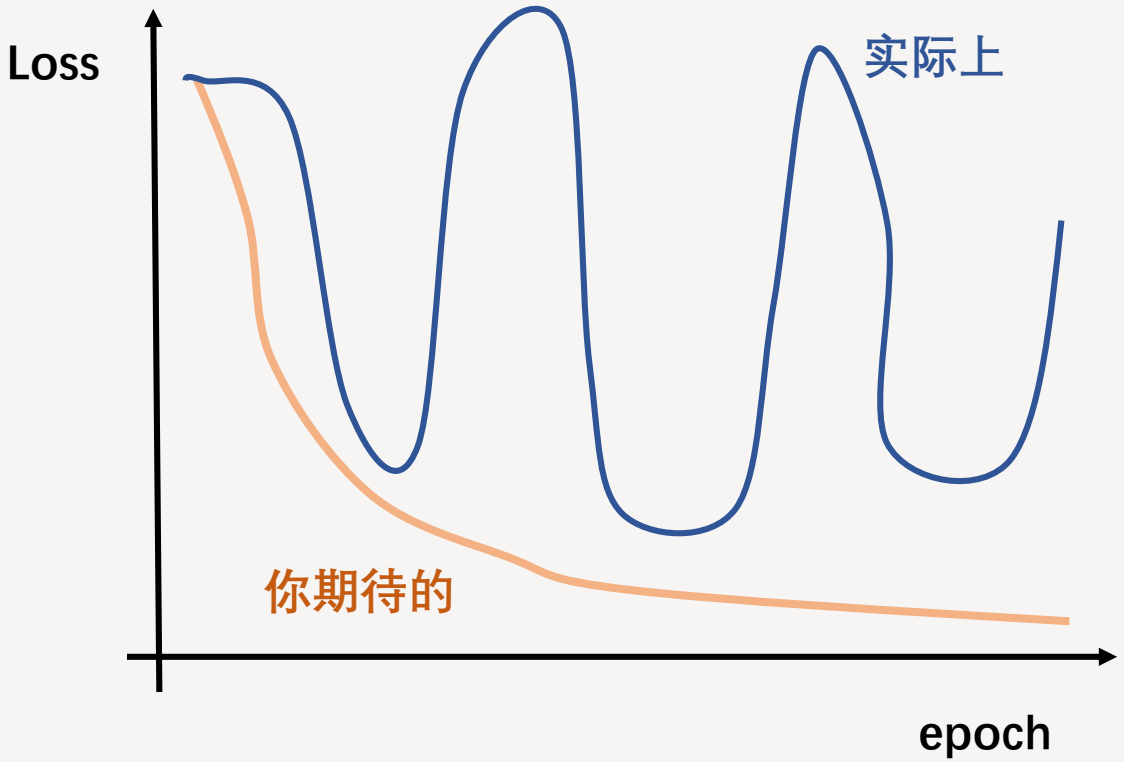
大家在网上看到的RNN可能长这个样子

RNN也可以是deep的!

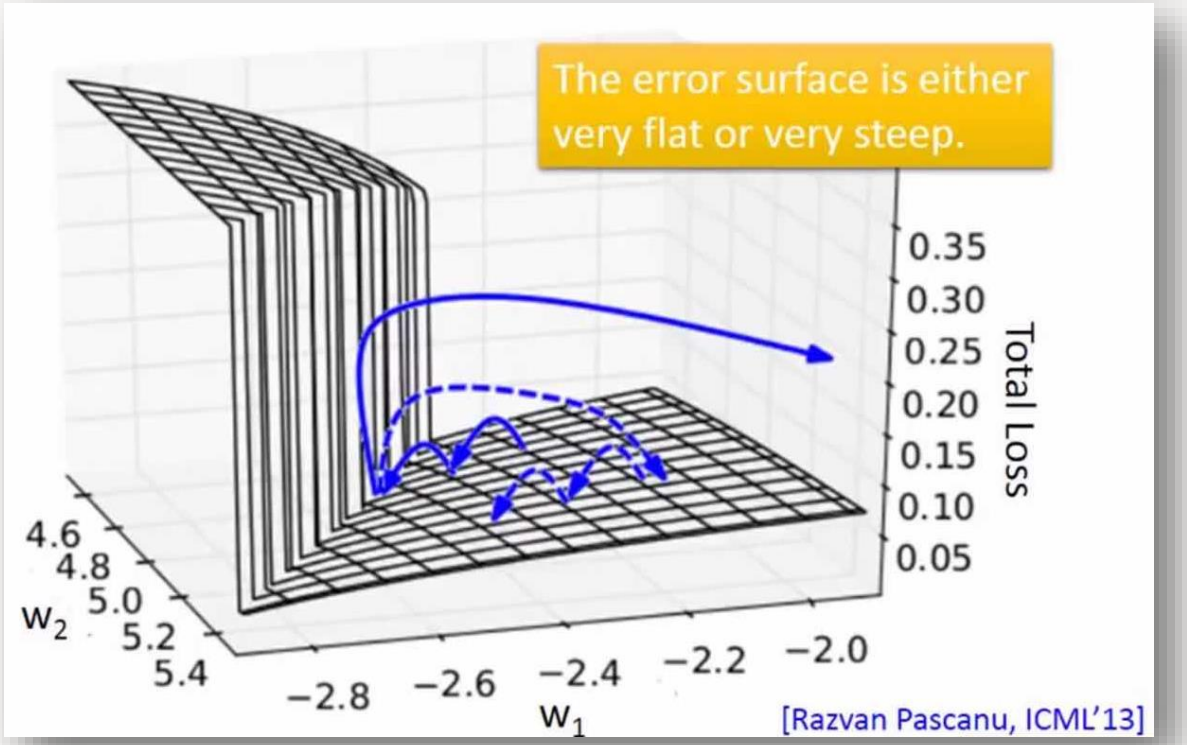


Recurrent Neural Network

但事实上RNN很难训练
Loss整体不随epoch增加而减少



Loss表面非常崎岖又陡峭
很容易出现梯度消失or梯度爆炸



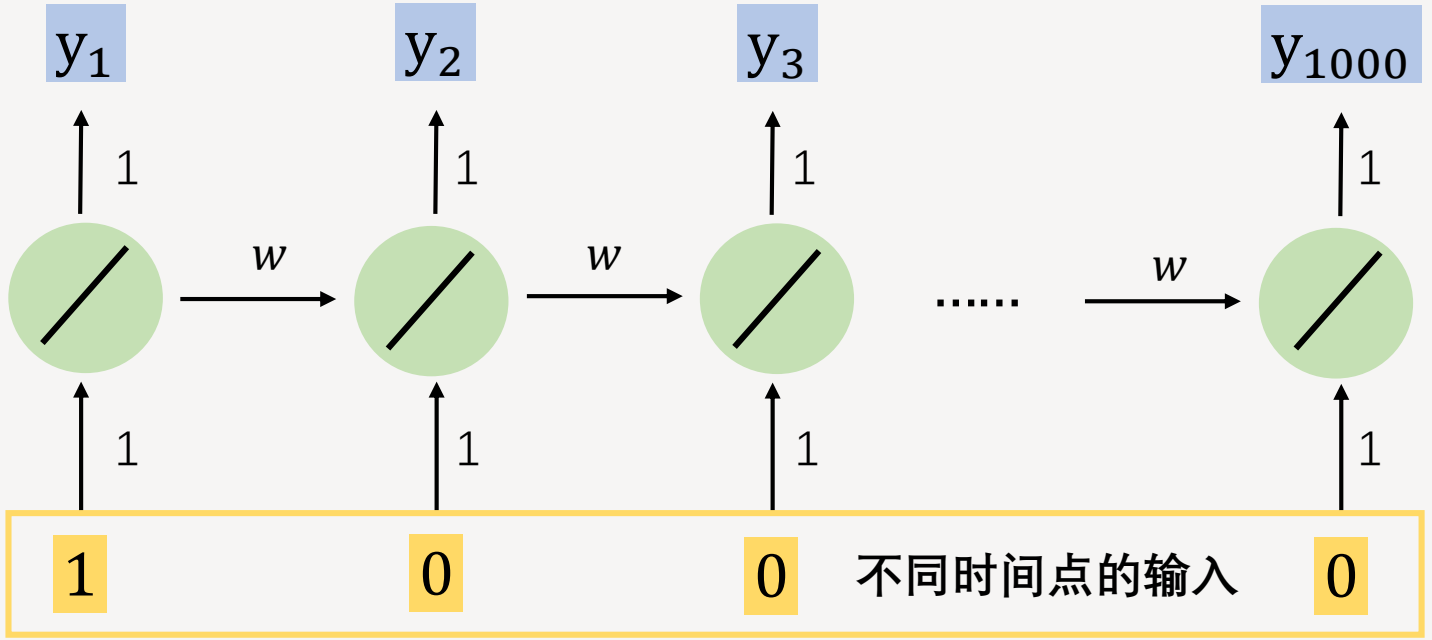
为什么RNN很难训练?

Why not RNN

我们选择LSTM

可以缓解梯度消失

有更灵活的记忆性



一点影响就会剧烈震荡

$w = 1 \rightarrow y_{1000} = 1$

$w = 1.01 \rightarrow y_{1000} \approx 20000$

$\frac{\partial L}{\partial w}$ 容易很大

设置小一点的learning rate?

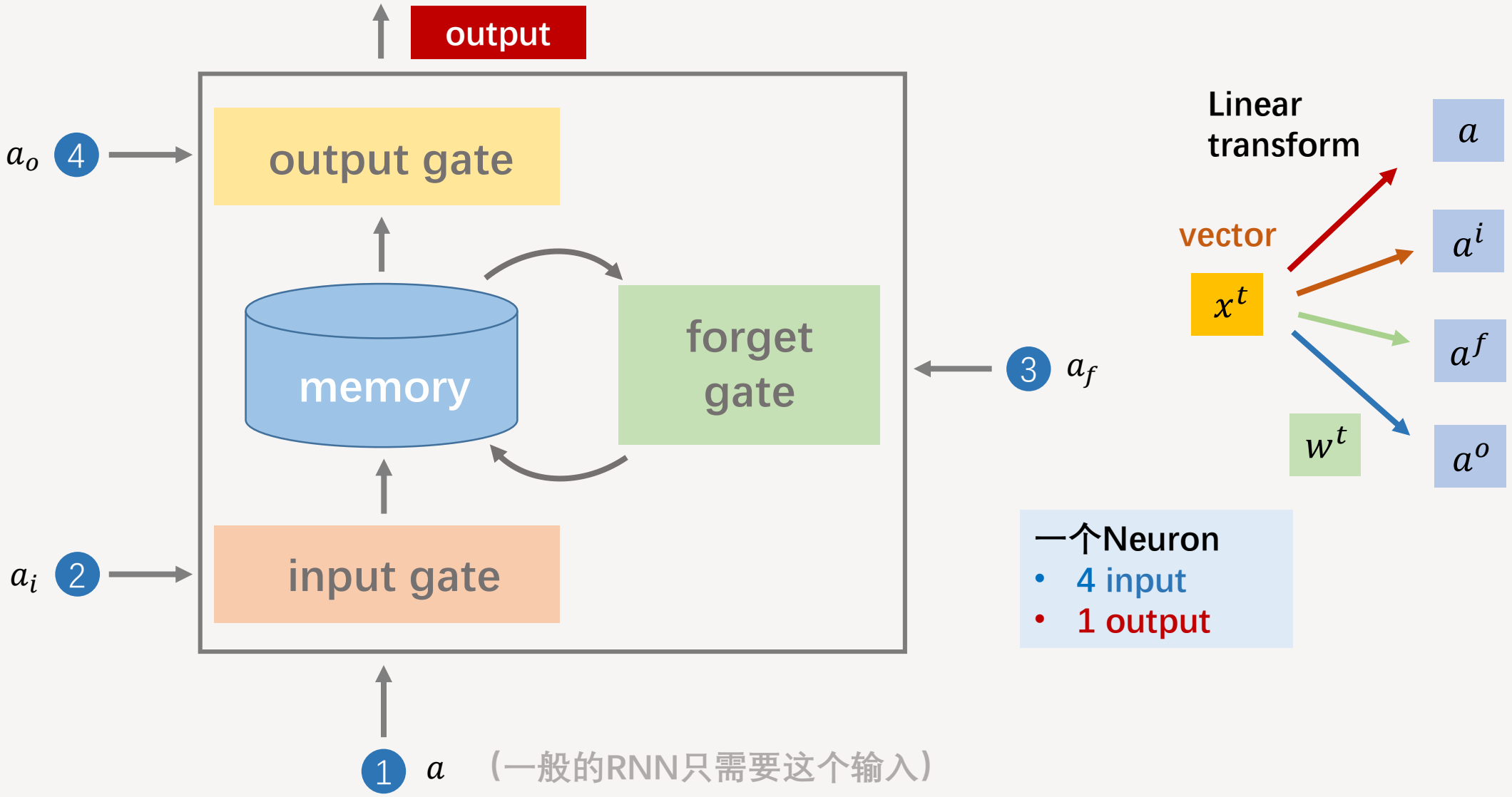
$w = 0.99 \rightarrow y_{1000} \approx 0$

$w = 0.01 \rightarrow y_{1000} \approx 0$

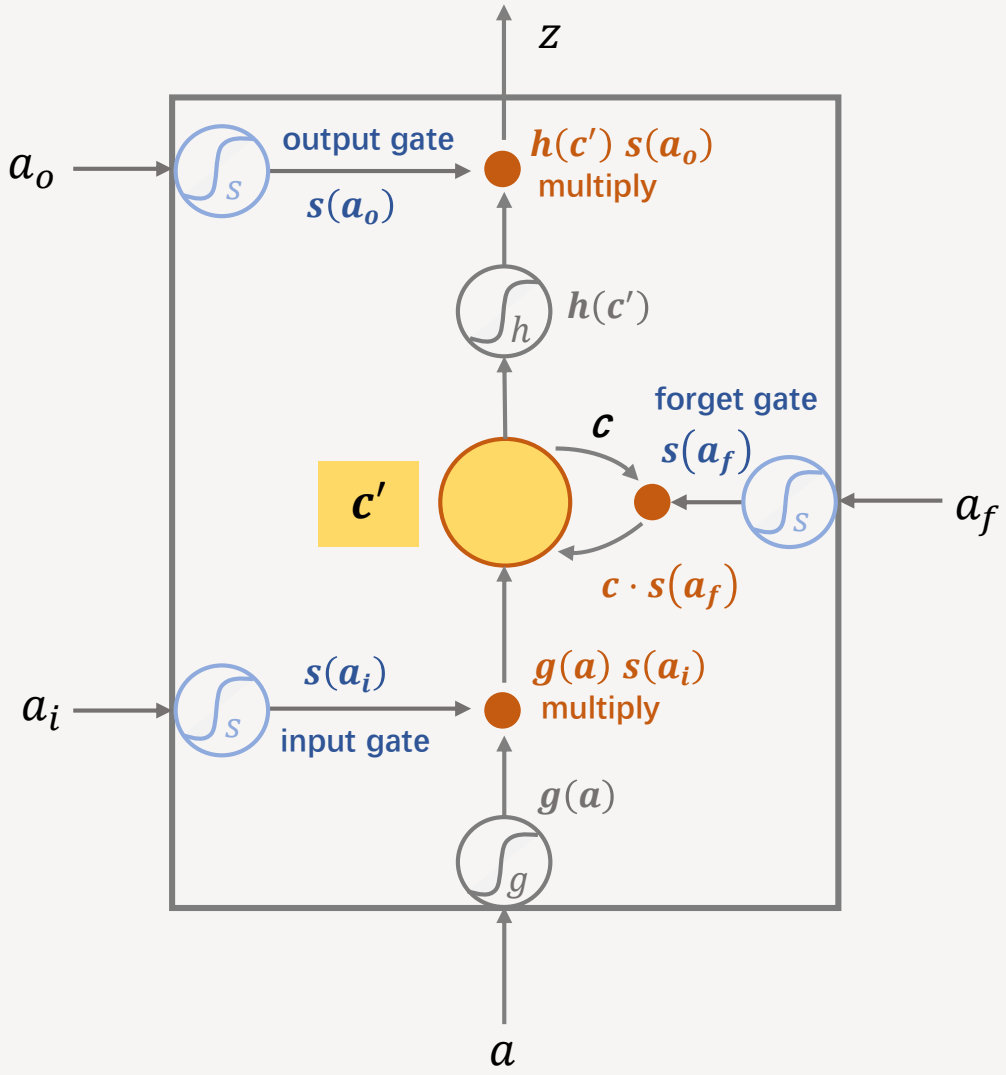
$\frac{\partial L}{\partial w}$ 突然变小

设置大一点的learning rate?

Long Short-Term Memory 长短期记忆



为了让大家更清楚，徒手画了一页LSTM



LSTM 比较长的短期记忆

增加的三个Gate，什么时候打开什么时候关闭
来控制信息量有多大概率被顺利传送过去
（权值是需要学习得到的）

sigmoid 函数: 生成0-1之间的值，表示概率

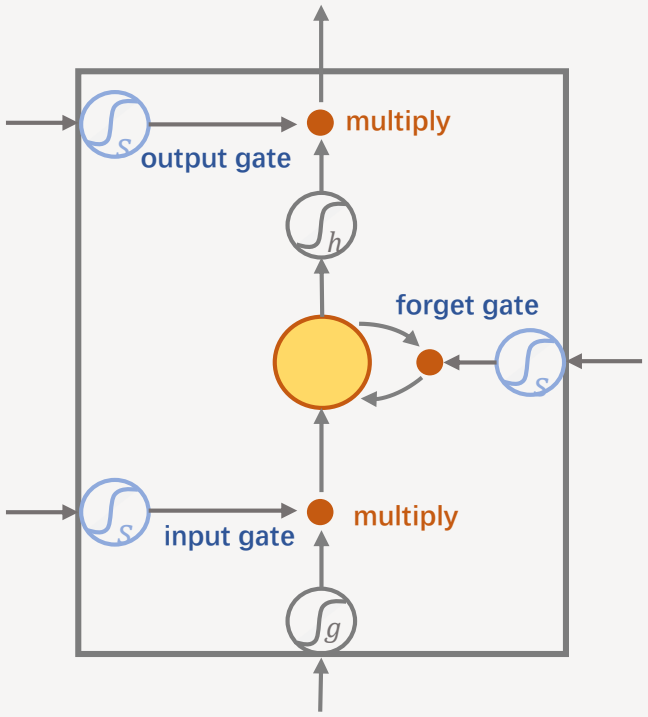
一般激活函数: 对传过来的值进行非线性变换

$$c' = g(a)\ s(a_i) + c \cdot s(a_f)$$

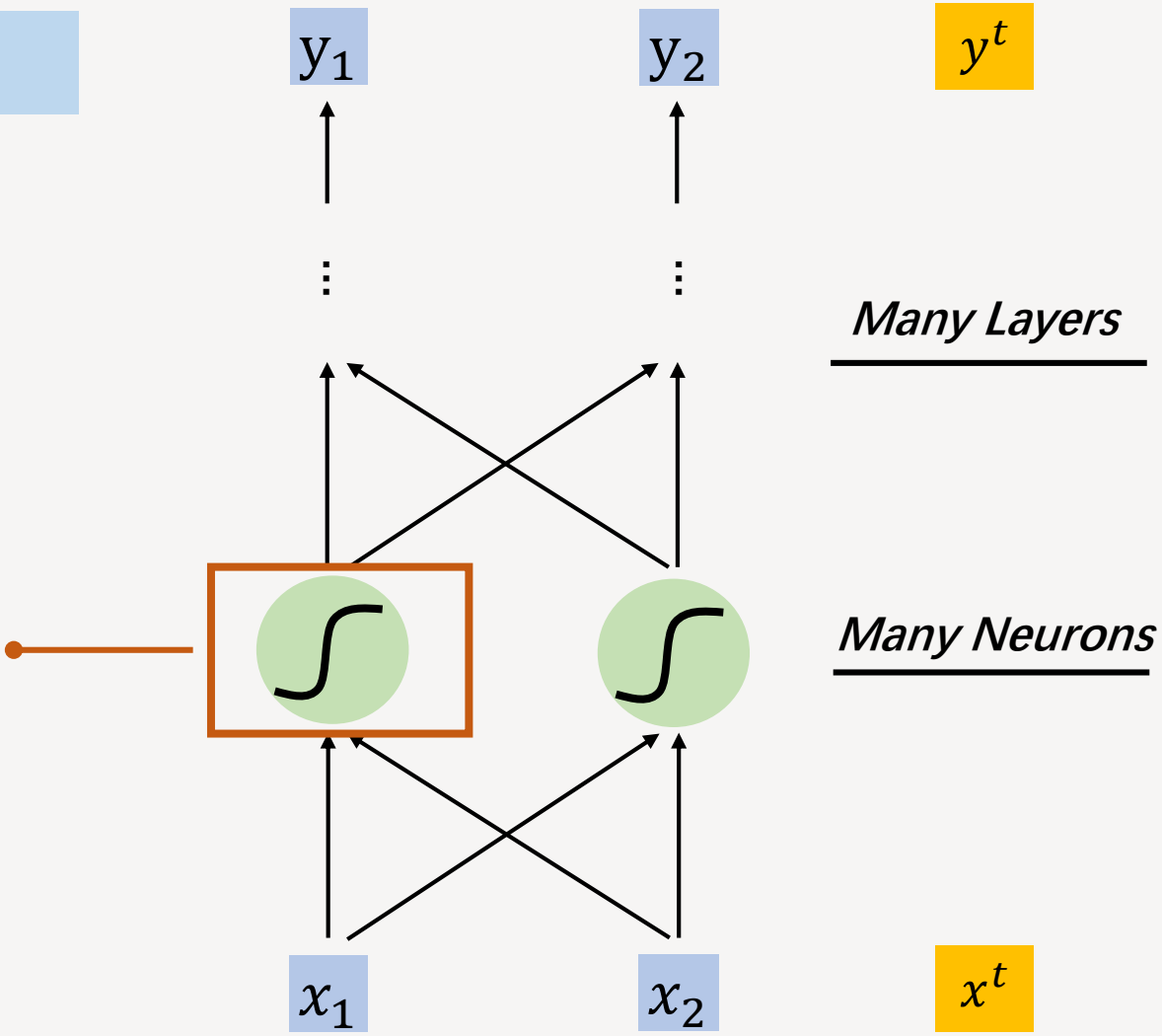
$$z = h(c')\ s(a_o)$$

LSTM vs Traditional Neural Network

Traditional Neural Network



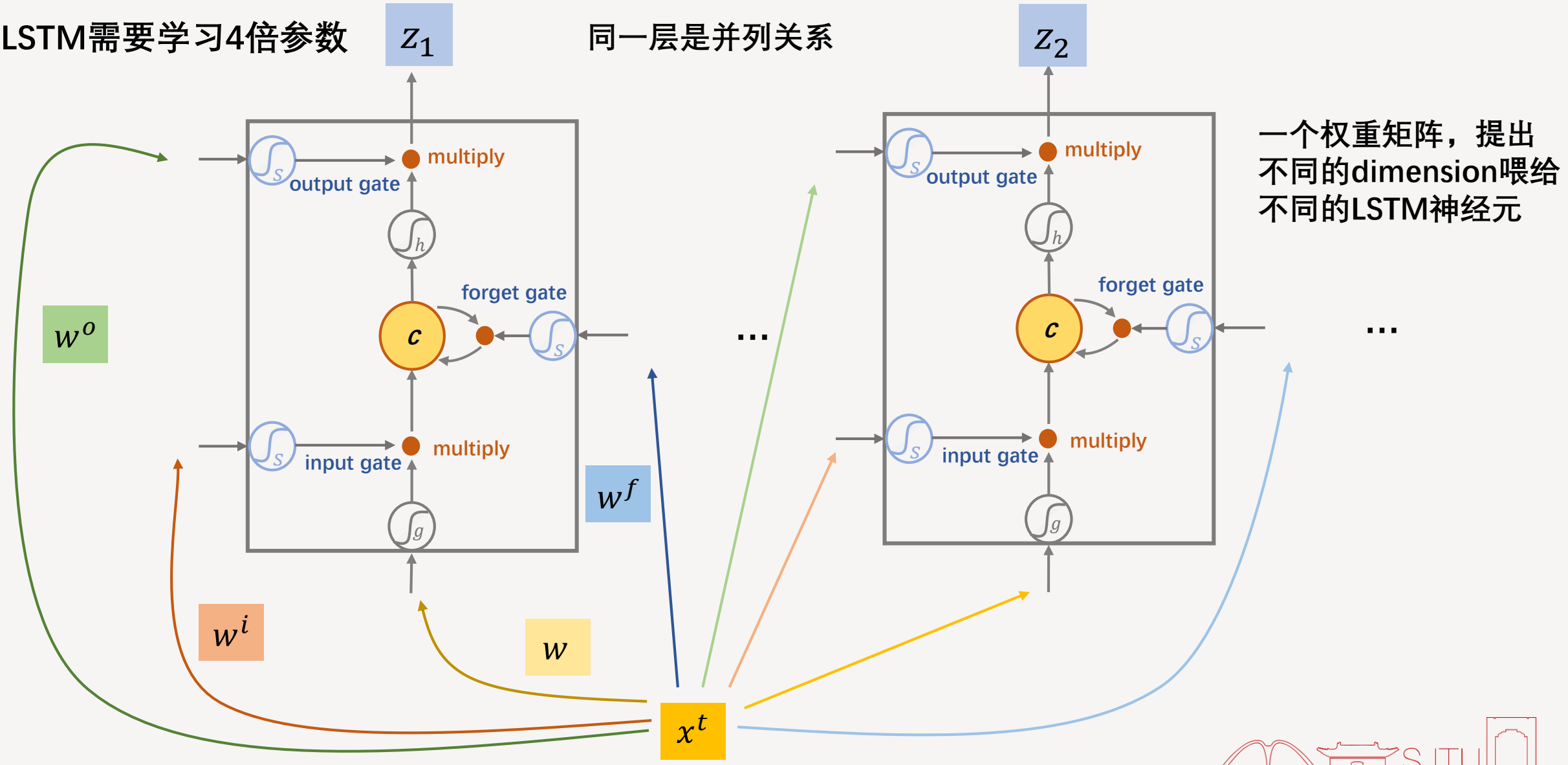
LSTM



LSTM vs Traditional Neural Network

LSTM需要学习4倍参数

同一层是并列关系

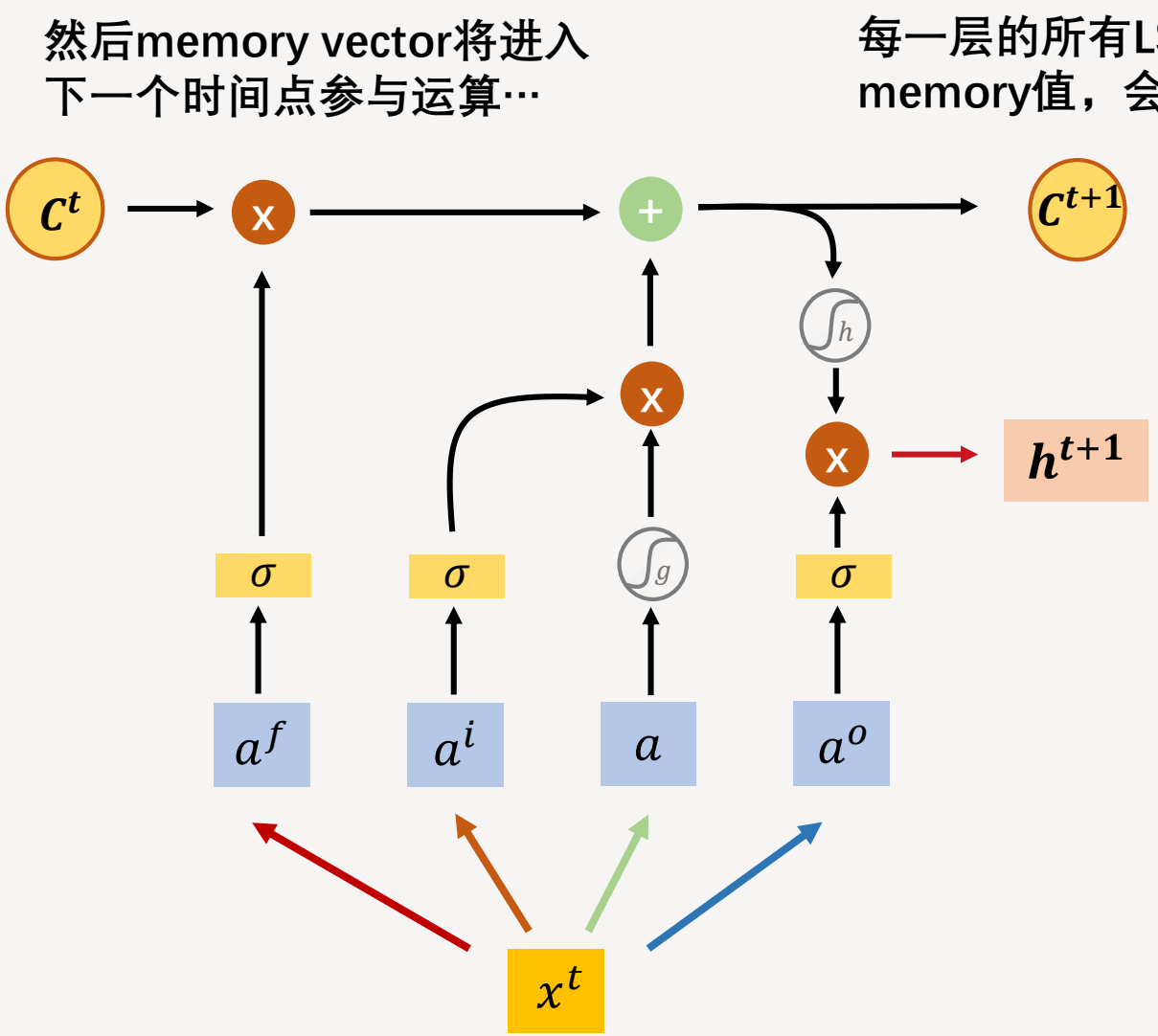


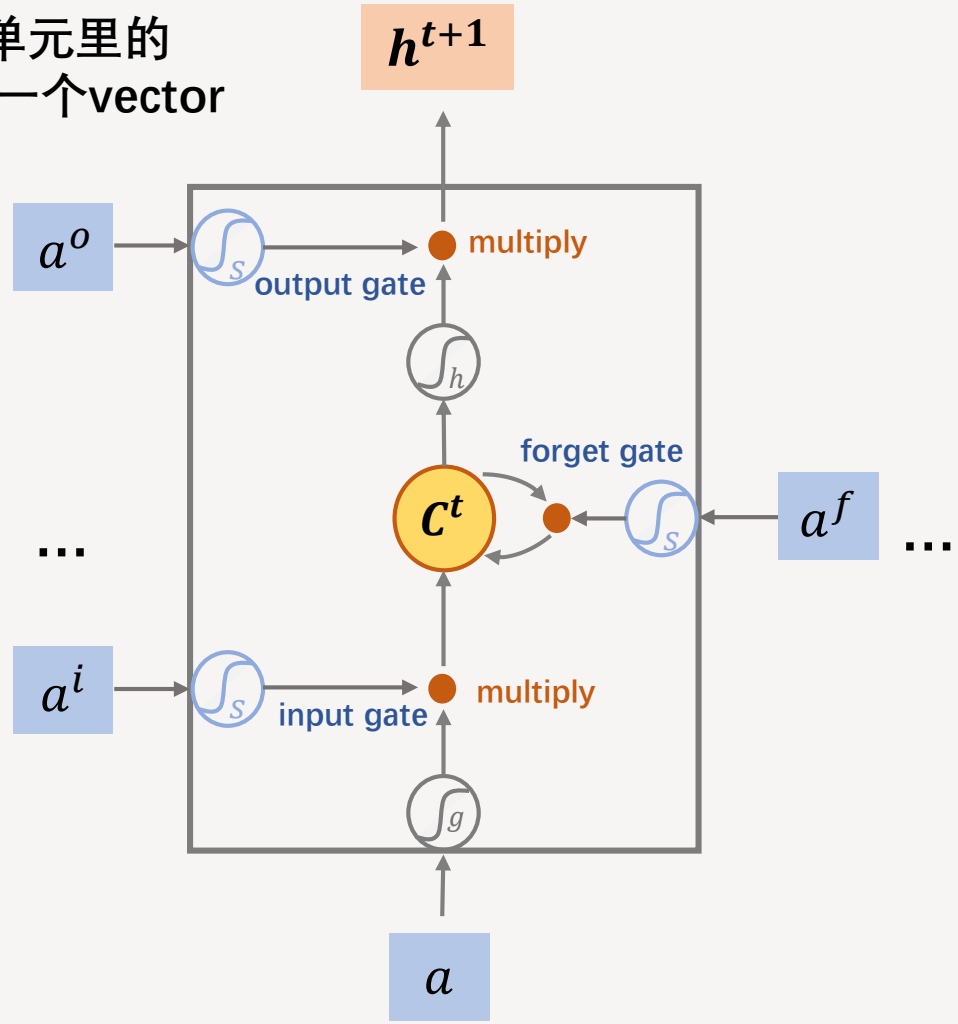
一个权重矩阵，提出不同的dimension喂给不同的LSTM神经元

回顾LSTM的流程

然后memory vector将进入
下一个时间点参与运算…

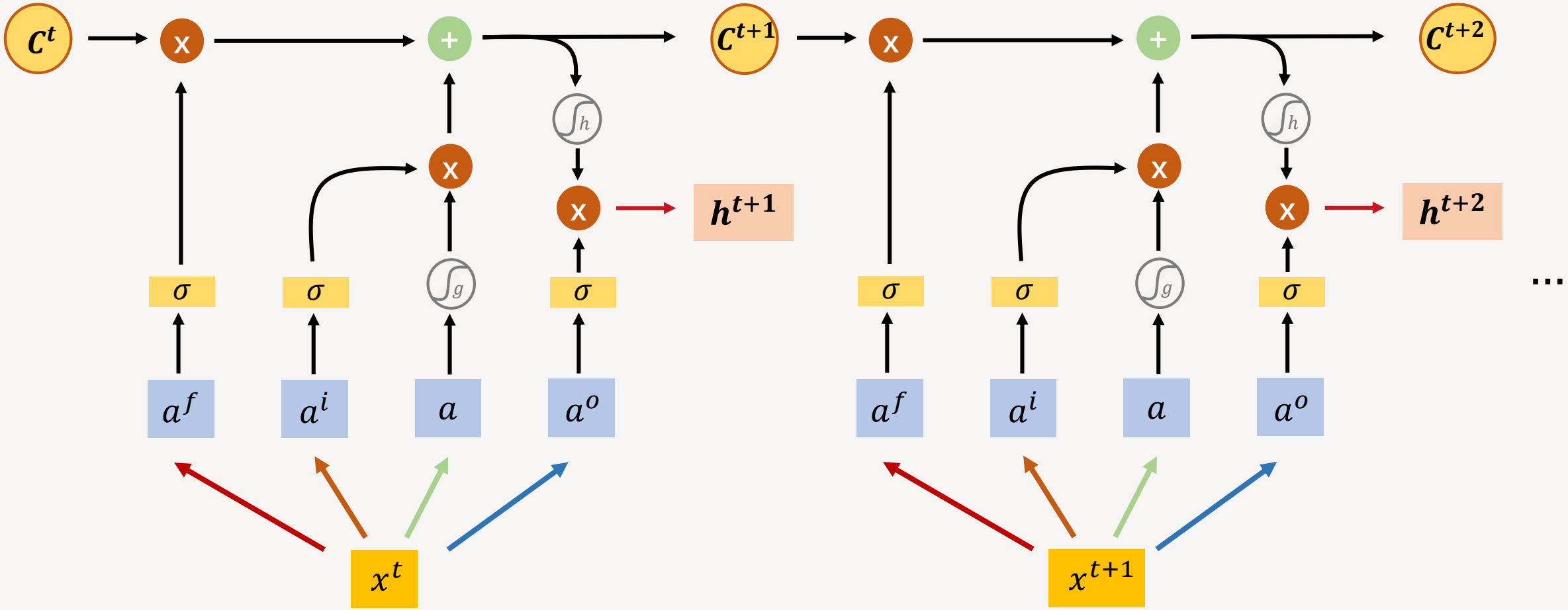
每一层的所有LSTM单元里的
memory值，会组成一个vector





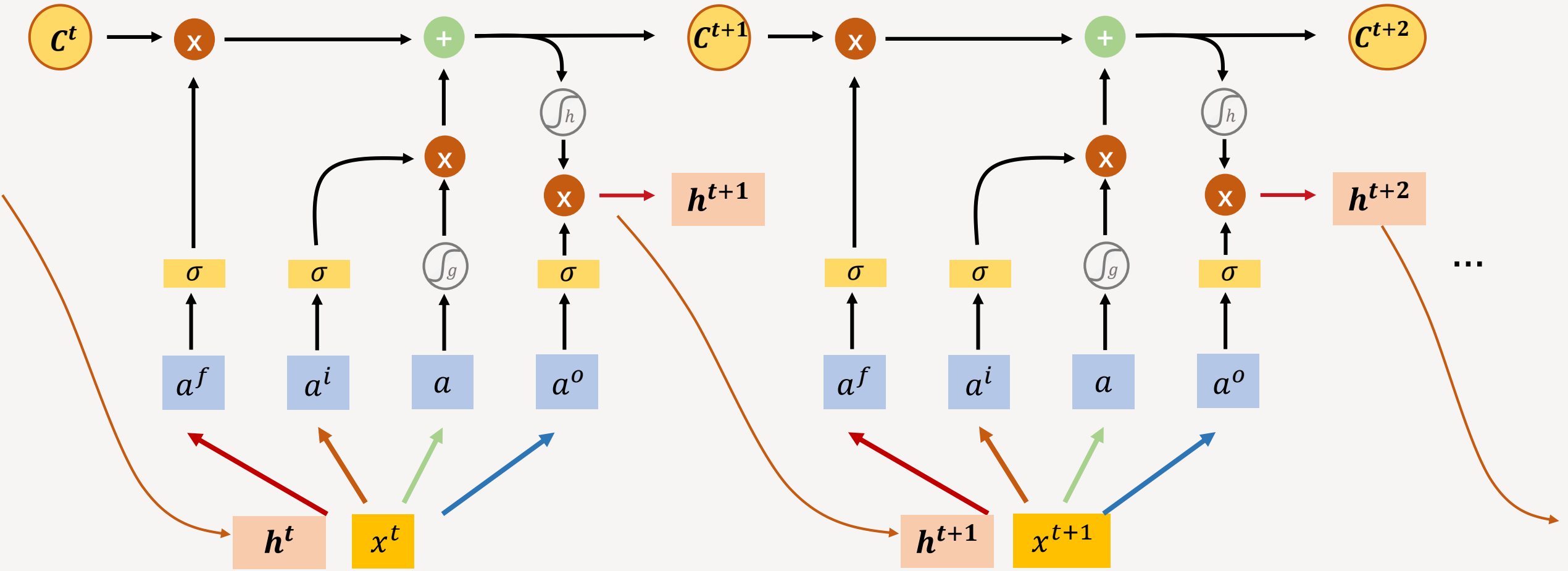
Long Short-Term Memory 长短期记忆

随着网络加深，继续迭代…



Long Short-Term Memory 长短期记忆

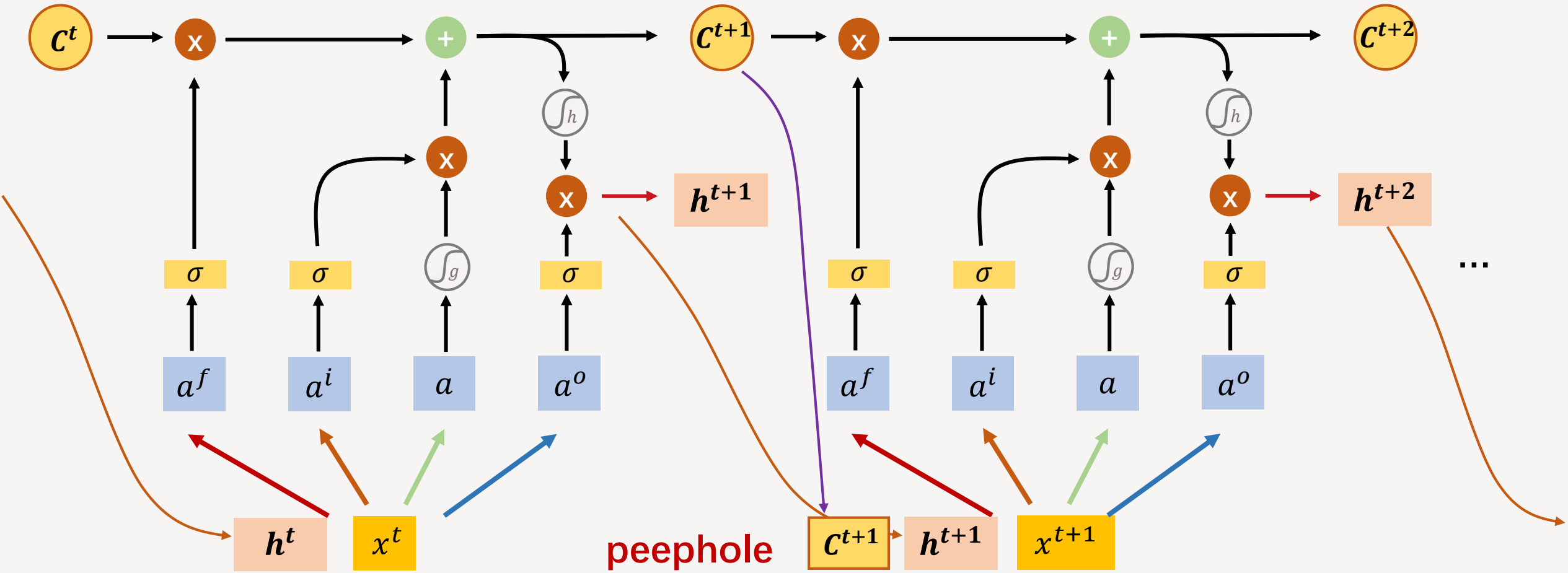
可能你已经觉得很复杂了，但…



Long Short-Term Memory 长短期记忆

甚至还有更复（bian）杂(tai)的情况…

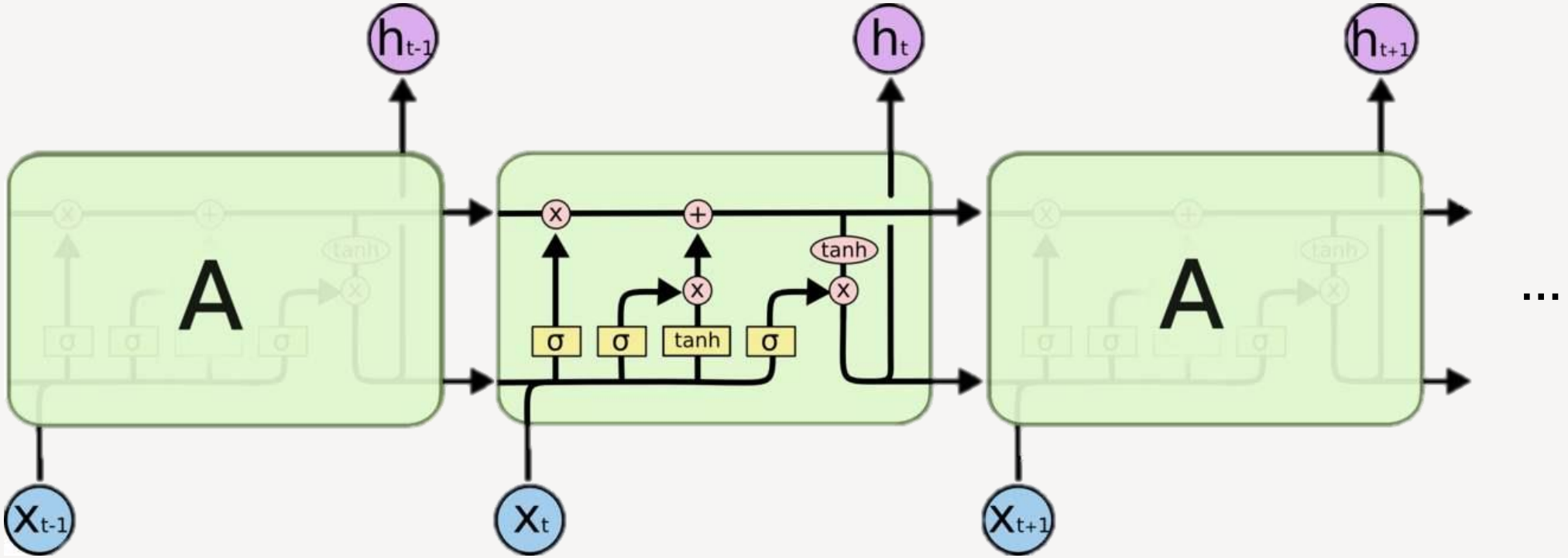
太难了！！



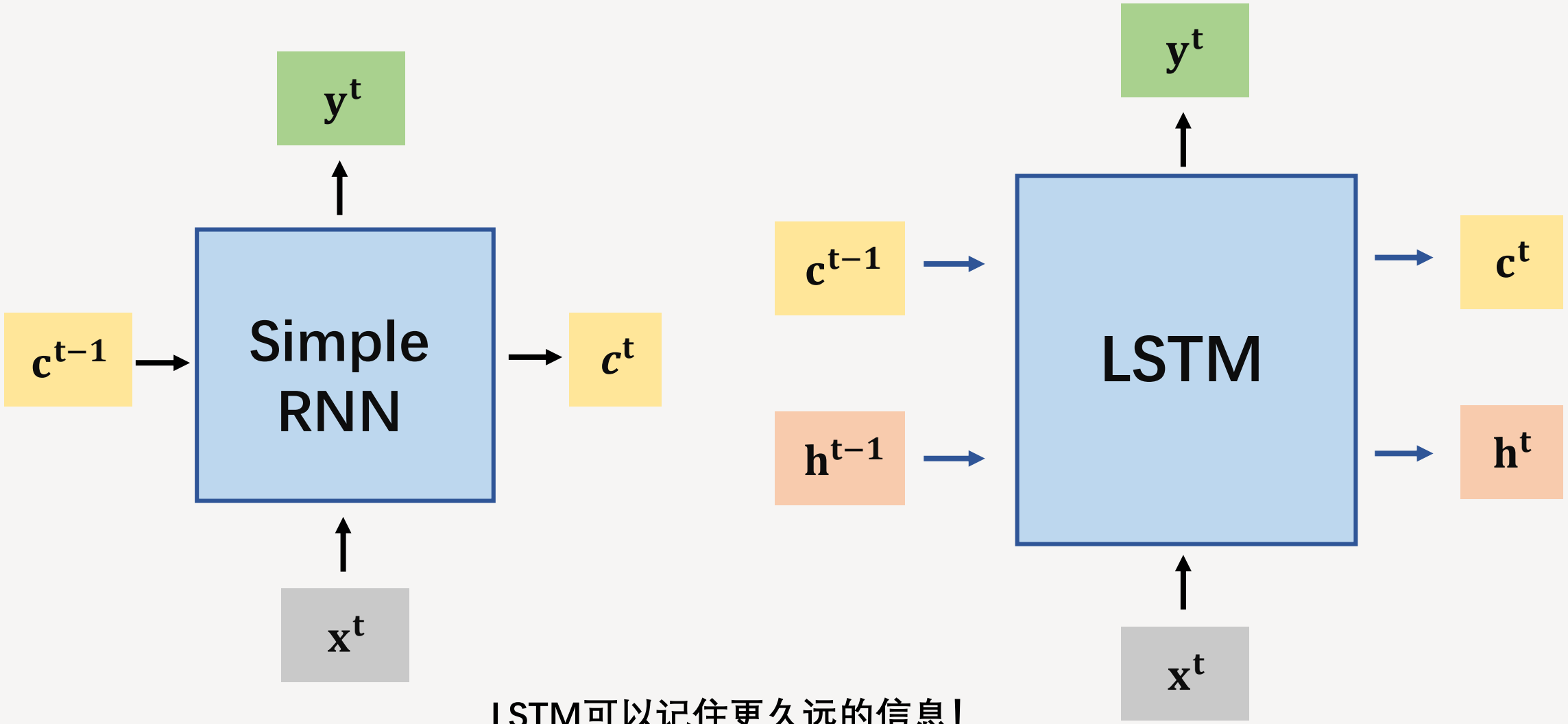
容易over fitting，设置一些trick

Long Short-Term Memory 长短期记忆

然后就会是你们在网上看到的这个示意图…



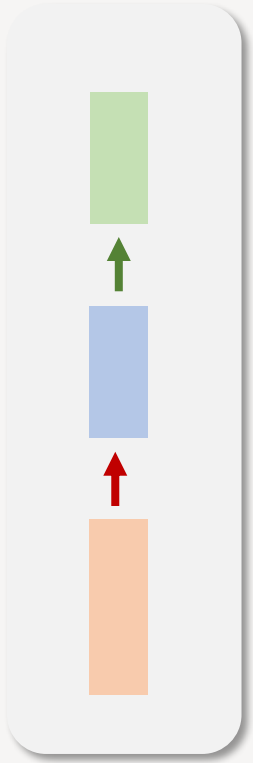
RNN vs LSTM



LSTM可以记住更久远的信息！

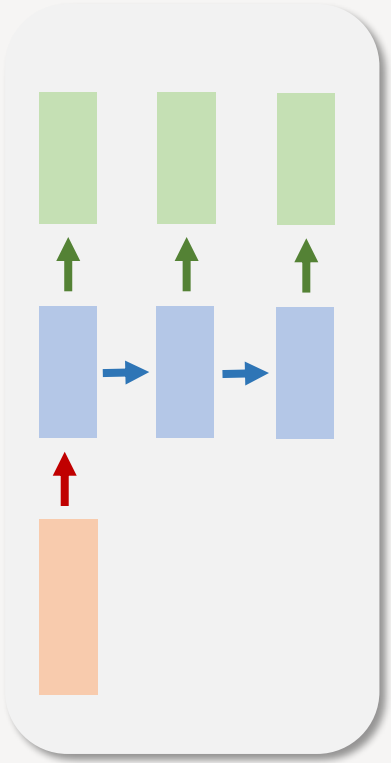
RNN/LSTM 的应用场景

1 to 1



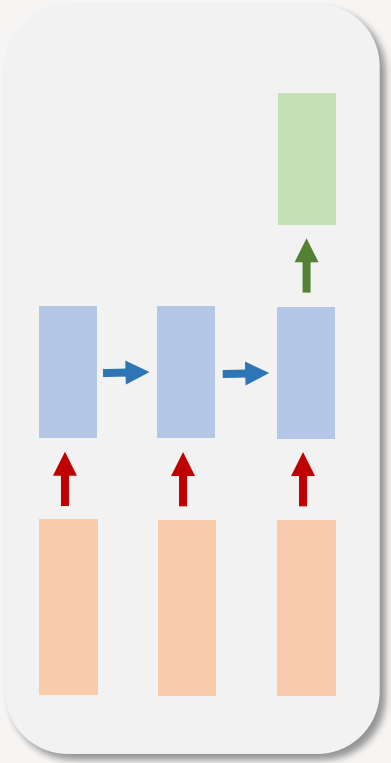
- 基本型

1 to Many



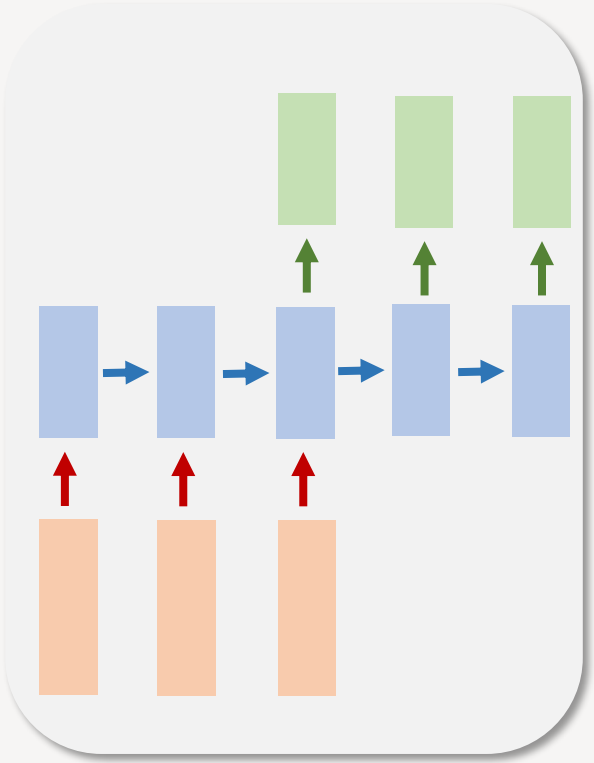
- 图片生成文字

Many to 1



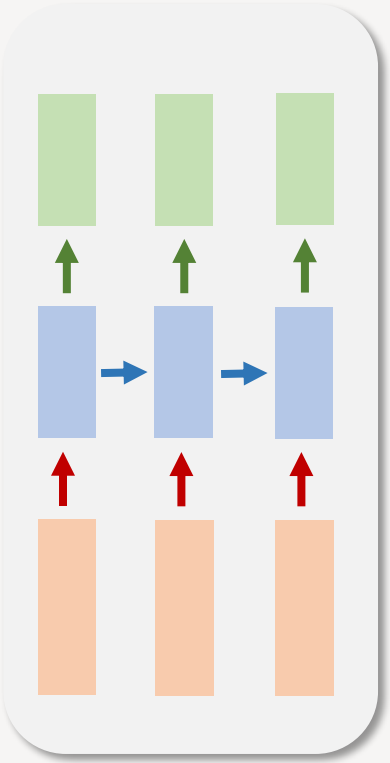
- 情感分析
- 关键词抽取

Many to Many



- 机器翻译
- 阅读理解

Many to Many



- 诗句
- 对联

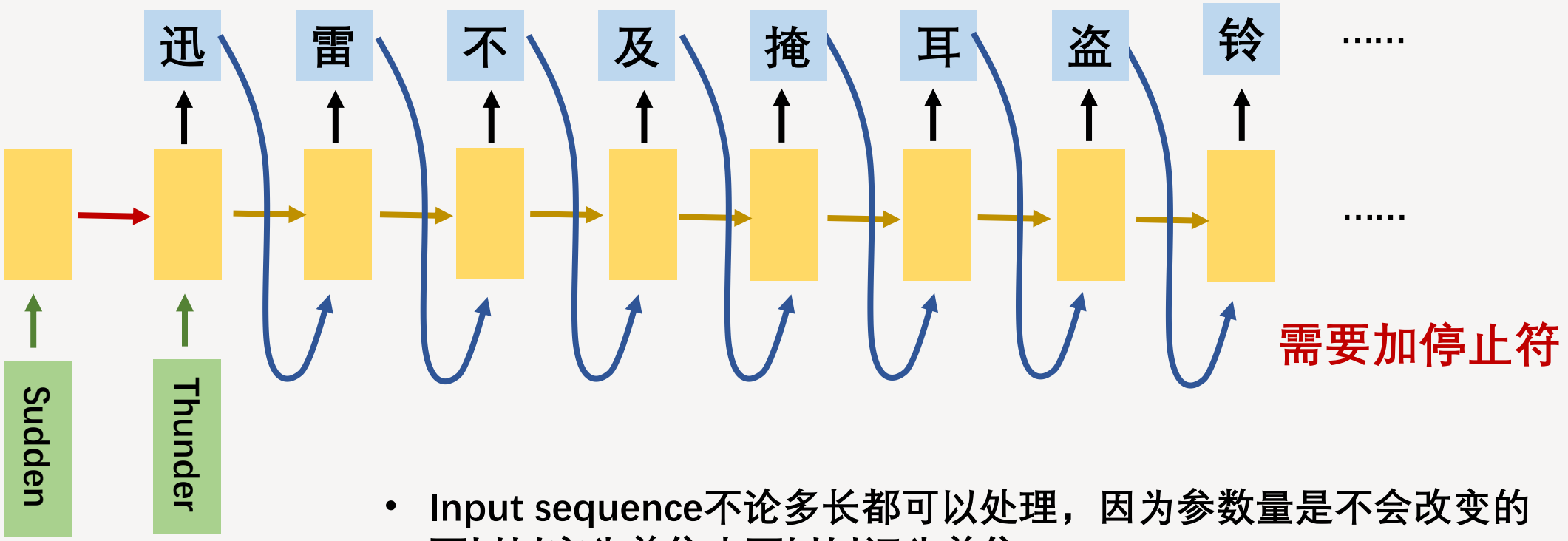
Seq2Seq 的应用场景

- 机器翻译（Neural Machine Translation）： NLP 中最经典的任务。
- 文本摘要：extractive 抽取式摘要和 abstractive 生成式摘要。前者是从一篇文档或者多篇文档中通过排序找出最有信息量的句子，组合成摘要；后者类似人类编辑一样，通过理解全文的内容，然后用简练的话将全文概括出来。基于 Seq2Seq+attention 模型的成功，有很多的工作都是用这种模式来做摘要任务，取得了一定的突破。
- 对话生成Chatbot：通过海量的数据来训练出一个智能体，回答任何开放性的问题。
- 此外还有很多其他的应用，如：诗词生成、音乐生成、风格转换、代码补全…

《Sequence to Sequence Learning with Neural Networks》： https://arxiv.org/pdf/1409.3215.pdf
《Learning Phrase Representation using RNN Encoder-Decoder for Statistical Machine Translation》： https://arxiv.org/pdf/1406.1078.pdf

Sequence 2 Sequence

Translation Input vector 和 Output vector都是序列，但不确定序列长度。

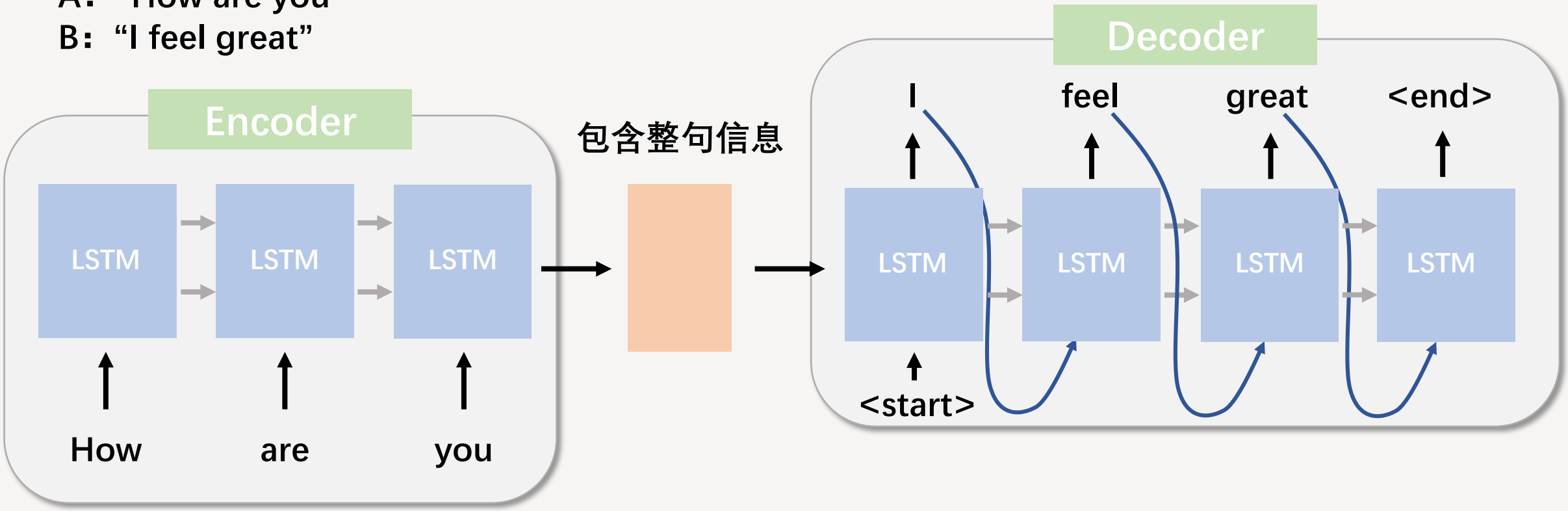


- Input sequence不论多长都可以处理，因为参数量是不会改变的
- 可以以字为单位也可以以词为单位
- 输出是一个分布（0，0.1，0.8，0.1）即下一个词的可能性

Sequence 2 Sequence

Chat-bot

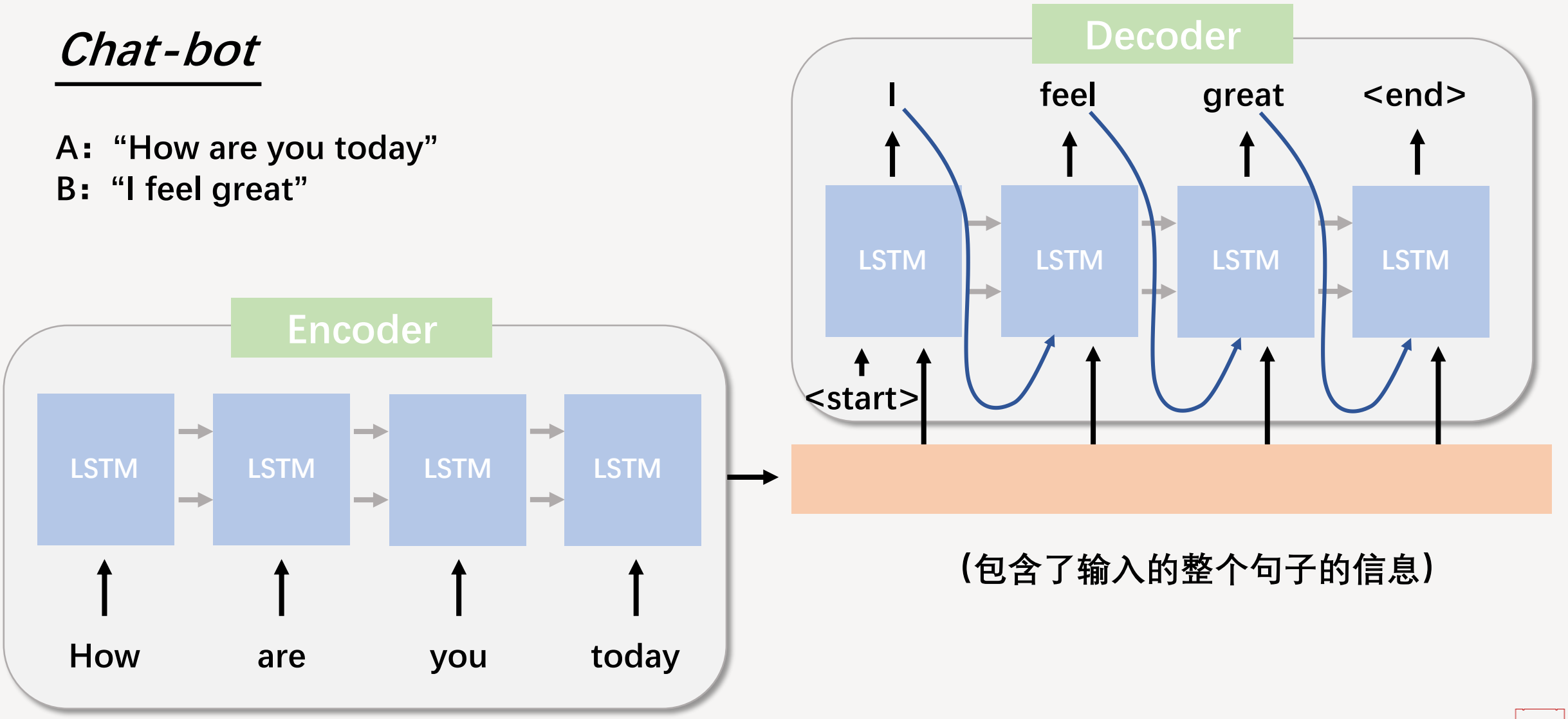
A：“How are you”
B：“I feel great”



Sequence 2 Sequence

Chat-bot

A：“How are you today”
B：“I feel great”

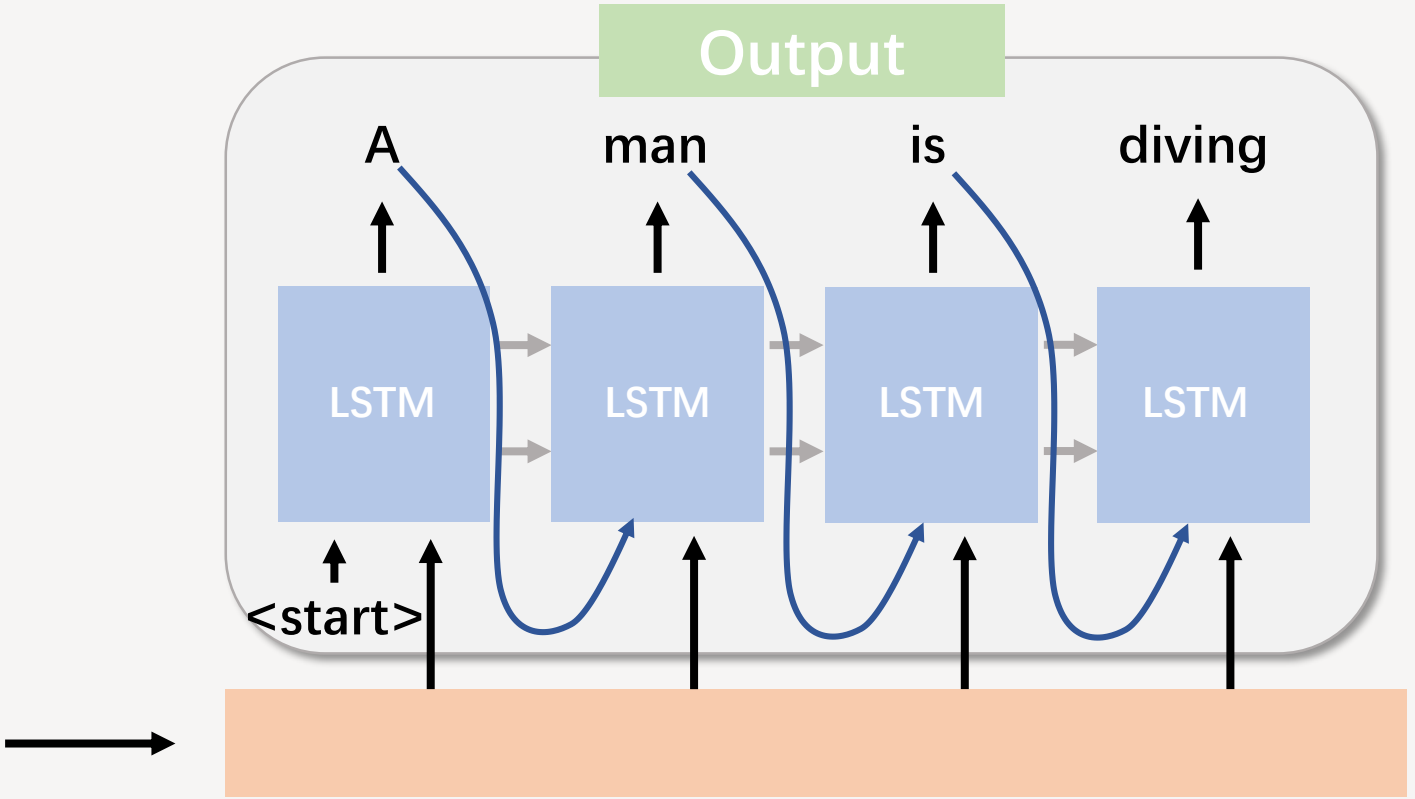


（包含了输入的整个句子的信息）

Sequence 2 Sequence

Caption Generation

(用CNN卷积提取Feature)



(包含了输入的整个图像的信息)

Sequence 2 Sequence

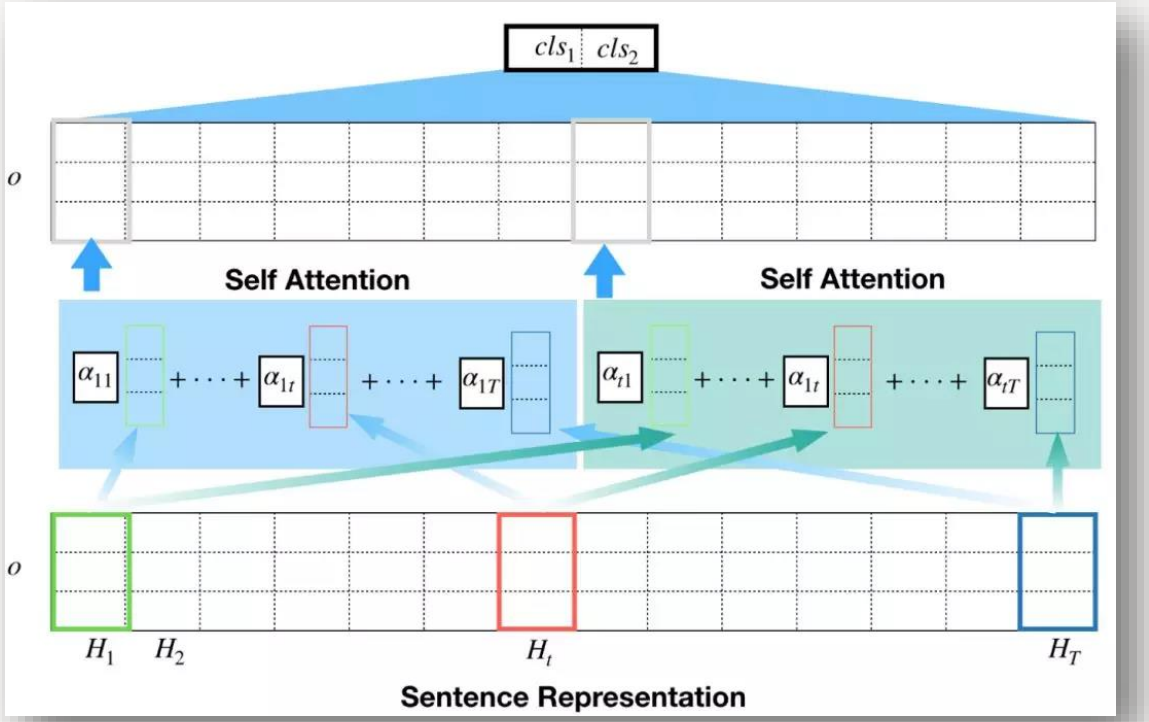
Encoder-Decoder框架虽然非常经典，但是局限性也非常大。

最大的局限性就在于：

编码器和解码器之间的唯一联系就是一个固定长度的语义向量。

- 可能无法完全表示整个序列的信息
- 先输入到网络的内容携带的信息会被后输入的信息覆盖掉，输入序列越长，这个现象就越严重。

这两个弊端使得在解码的时候解码器一开始就没有获得输入序列足够多的信息， 那么解码的准确度也就不高了。



Self-Attention 注意力机制

Thanks

学生创新中心：肖雄子彦