循环神经网络

演讲老师 肖雄

目录

目录

1.概述

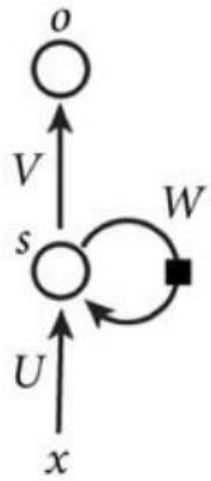
什么是RNN?

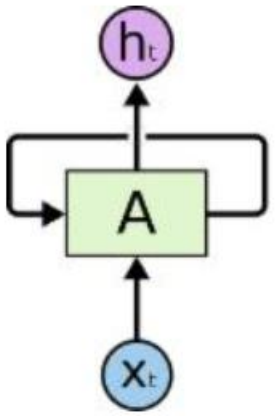
循环神经网络（Recurrent Neural Network, RNN）

百度百科：是一类以序列（sequence）数据为输入，在序列的演进方向进行递归（recursion）且所有节点（循环单元）按链式连接的递归神经网络（recursive neural network）。

RNN是神经网络的一种;

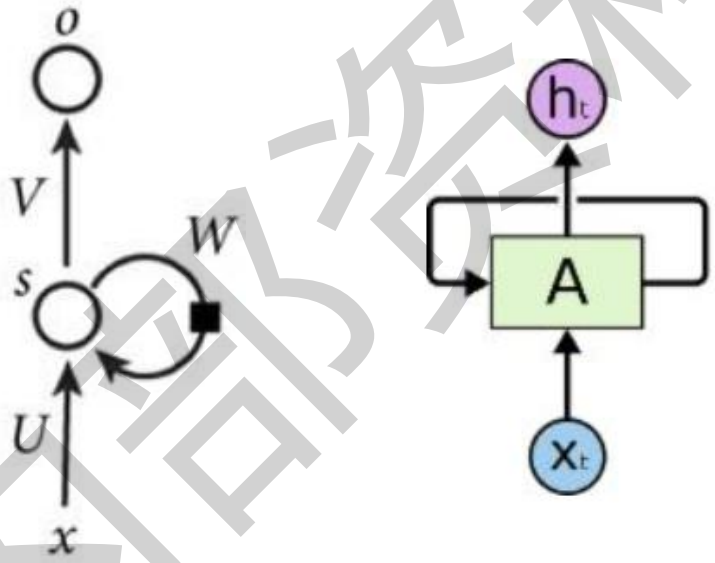
RNN有两种常见形式：
双向循环神经网络（Bidirectional RNN, Bi-RNN）；
长短期记忆网络（Long Short-Term Memory networks，LSTM）

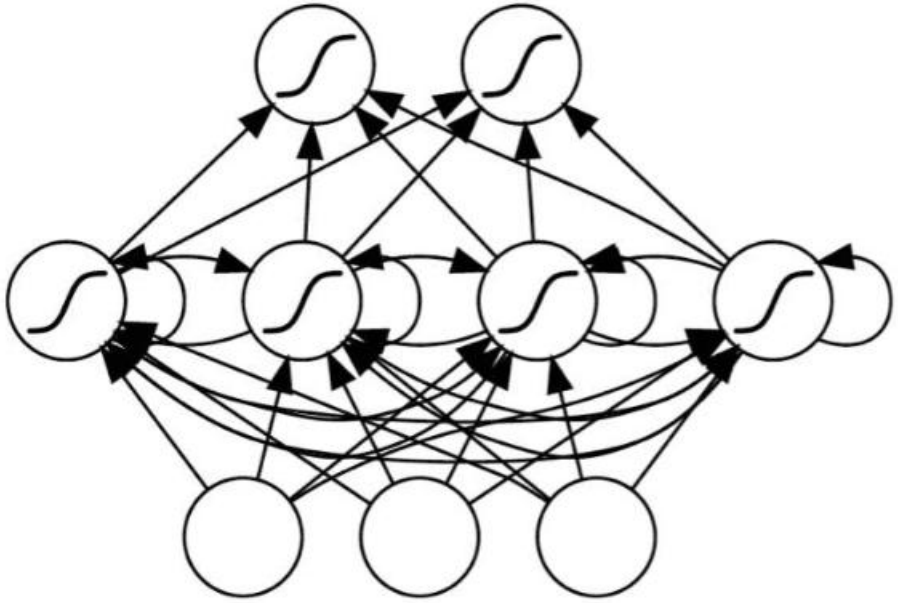


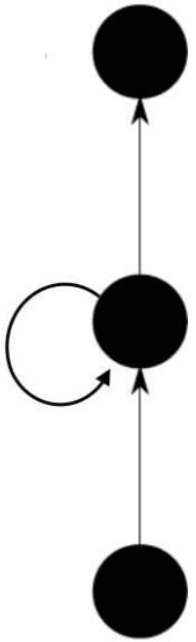


RNN常见图形标识

论文中RNN的常见图形标识有：







RNN的应用领域

自然语言处理：
自然语言数据是典型的序列数据，RNN的专长便是处理序列数据，因此在NLP领域大量使用，比如“机器翻译”、“文本分类”等等。

语音识别：
RNN可被应用于端到端（end-to-end）语言建模，语音合成、语音转换、识别语音并转成文字。

计算机视觉：
RNN结合CNN用于OCR技术，实现图片文字识别；视频行为识别。

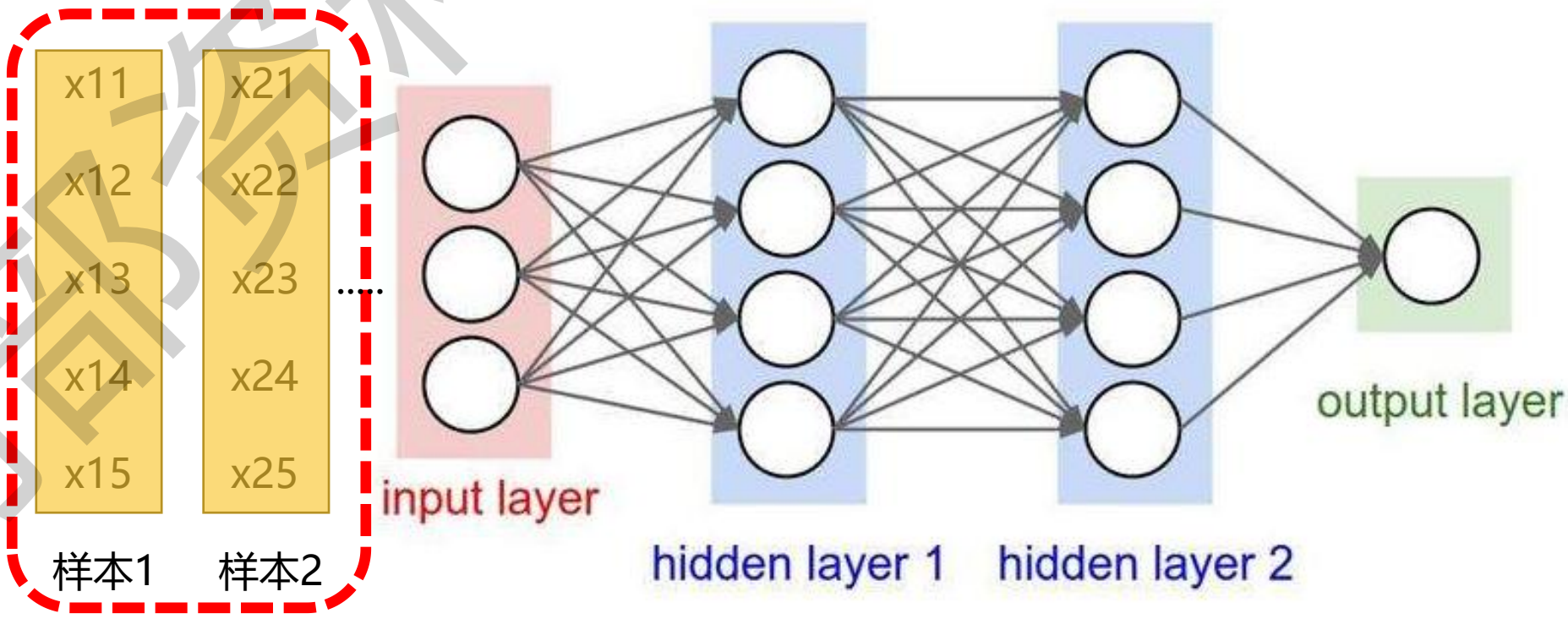
其他领域：
DNA序列分析、股票预测、天气预报等。

2.序列数据

全连接网络的输入

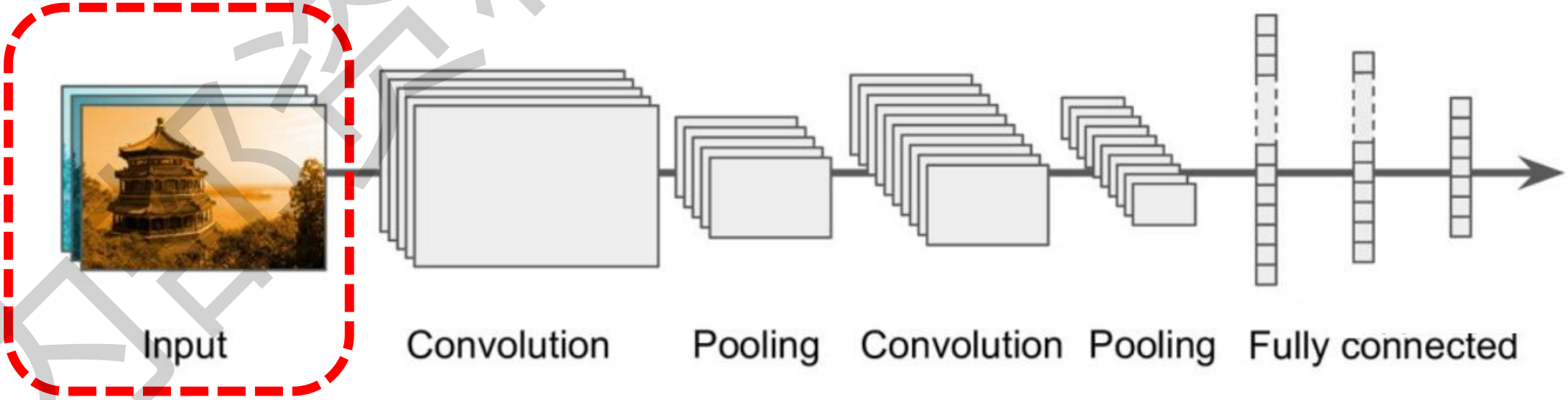
全连接神经网络：

输入的样本之间是相互独立的



卷积神经网络的输入

输入的图片之间是相互独立的



有关序列数据的两个例子：

我一直想去大连旅游，可是一直没机会，等有时间了，我一定要去“_____”

亲！你不会填“海南”吧！

我很喜欢吃香蕉，我每天都要吃一根“_____”

亲！你不会填“老冰棒”吧？

序列数据举例：

疫情数据

股票价格

天气数据

样本间存在顺序关系，每个样本和它之前的样本存在关联

专门用来处理序列数据的组件（RNN、LSTM等）

- 针对对象：序列数据。
  - 例如
    - 文本，是字母和词汇的序列；
    - 语音，是音节的序列；
    - 视频，是图像的序列；
    - 气象观测数据；
    - 股票交易数据等等
- 核心思想：
  - 通过神经网络在时序上的展开，我们能够找到样本之间的序列相关性。

- **主要应用领域**
  - **语音识别**
  - **音乐合成**
  - **情感分类**
  - **DNA序列分析**
  - **机器翻译**
  - **命名实体识别**
  - **…**

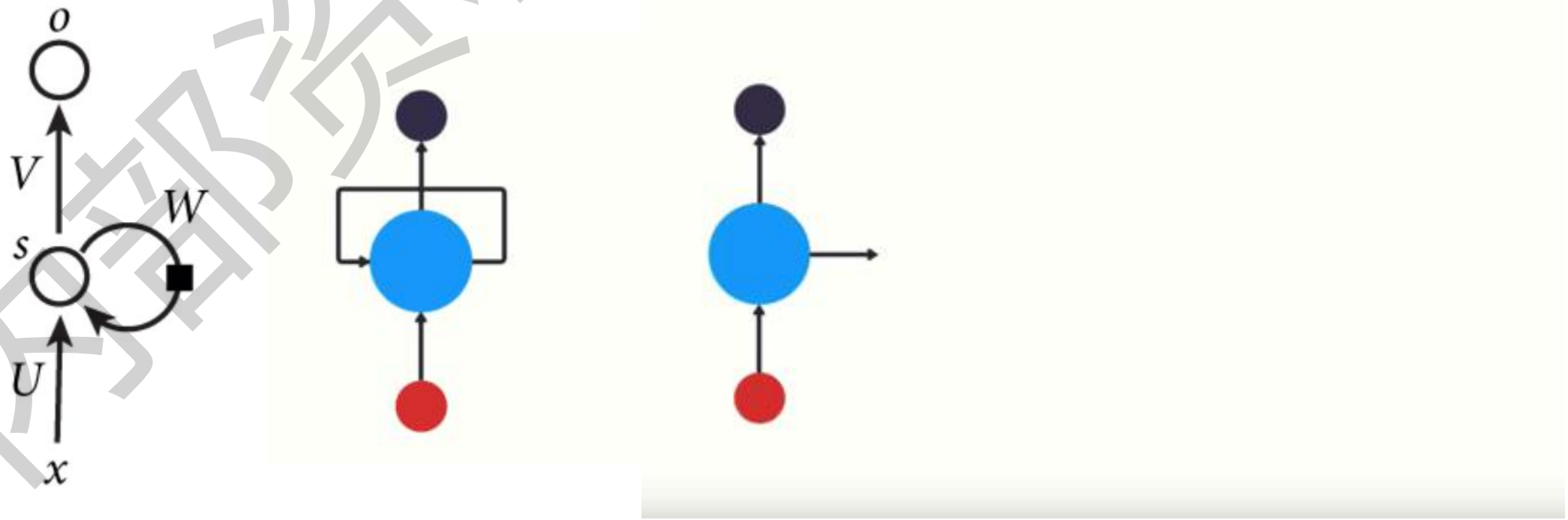
3.RNN结构原理

$$y = f(Wx + b)$$

最基础的神经网络单元

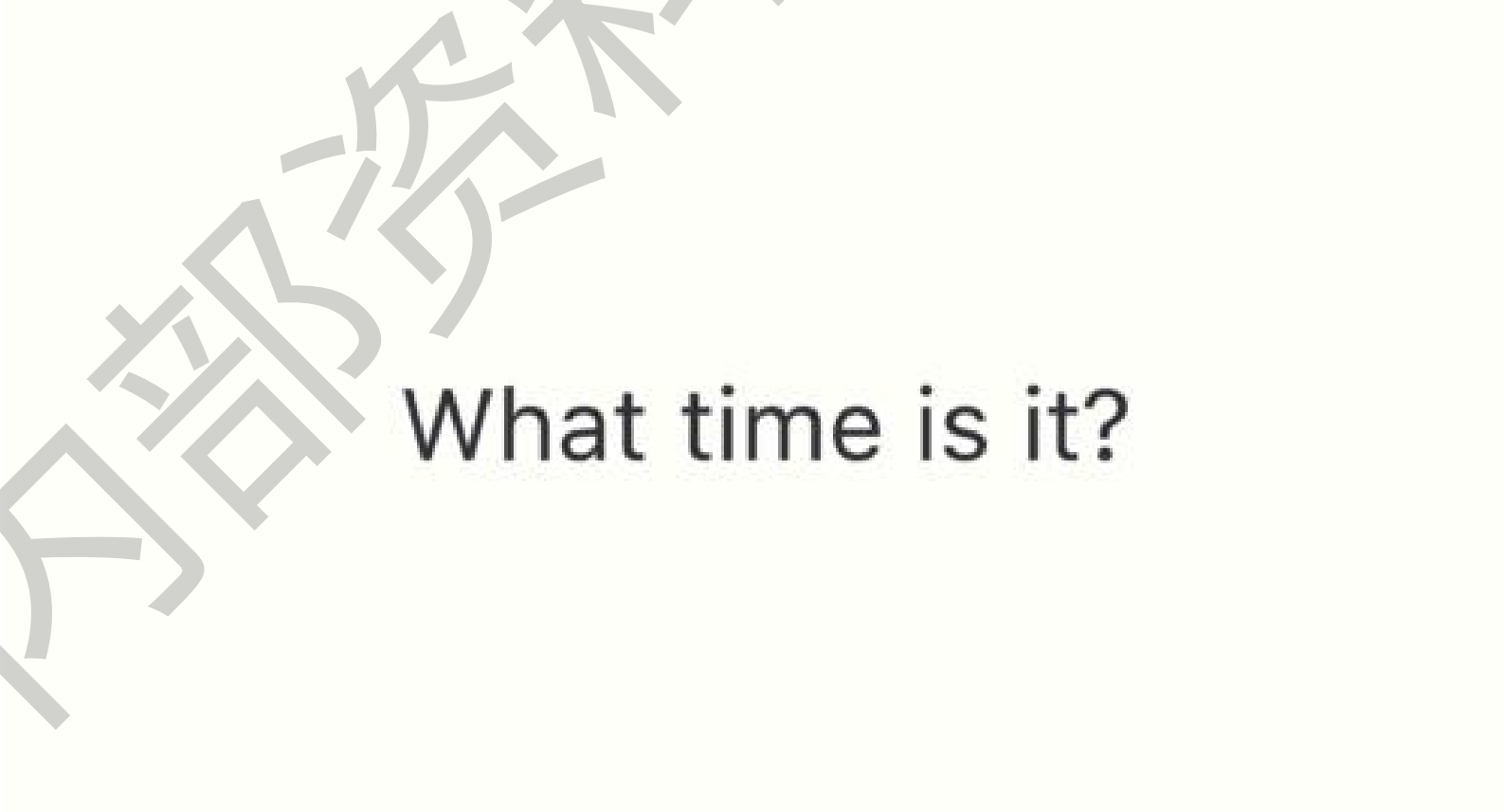
RNN结构

一个最简单的循环神经网络结构，（右图为动态图），它由输入层、一个隐藏层和一个输出层组成



RNN工作-动画演示-1

首先，我们将句子分解为单个单词。 RNN按顺序工作，我们一次输入一个字。

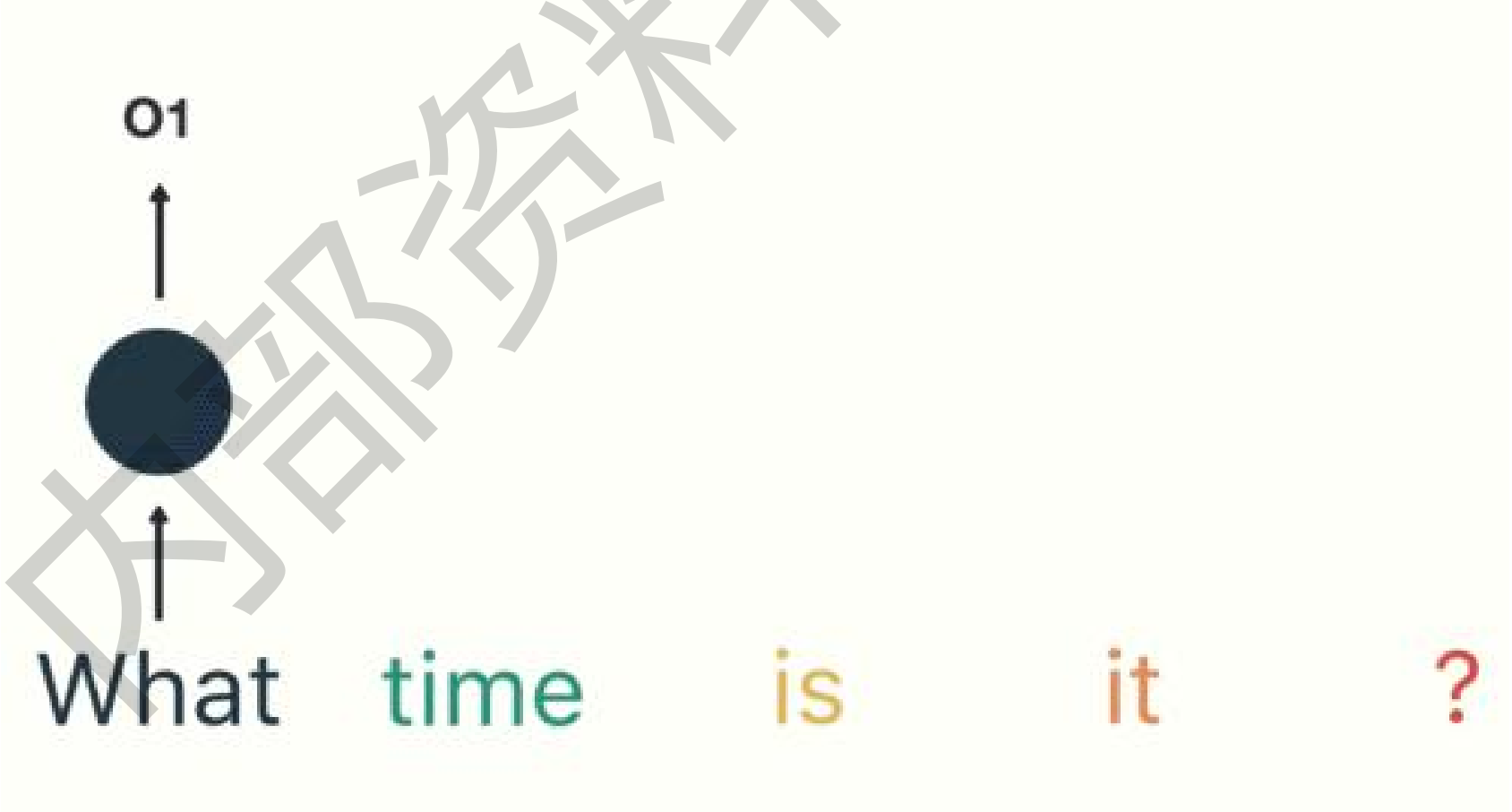


RNN工作-动画演示-2

第一步是将“What”输入RNN。 RNN编码“ What ”并产生输出。

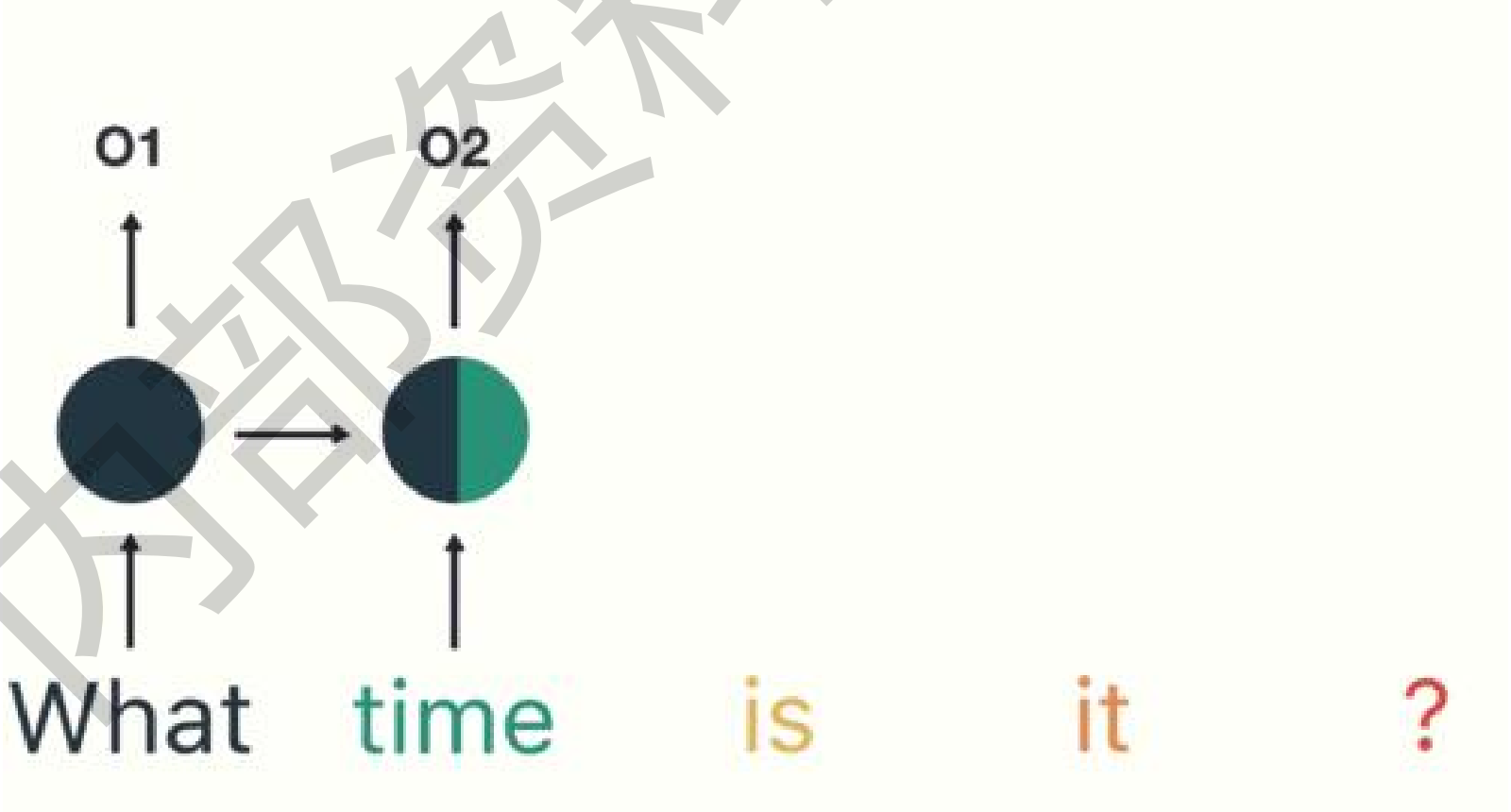
RNN工作-动画演示-3

下一步，我们提供单词“time”和上一步中的隐藏状态。 RNN现在有关于“What”和“time”这两个词的信息。



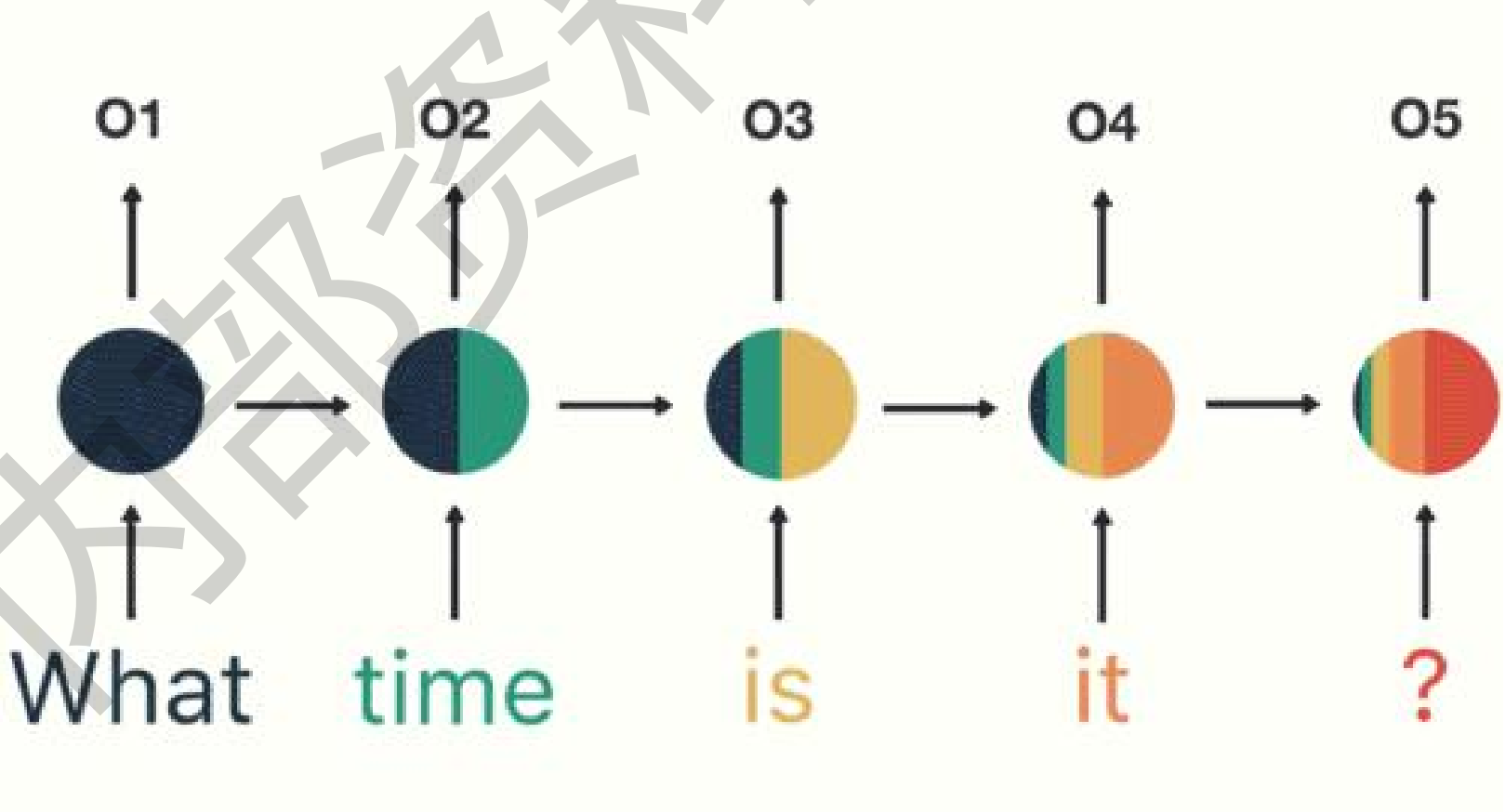
RNN工作-动画演示-4

我们重复这个过程，直到最后一步。 你可以通过最后一步看到RNN编码了前面步骤中所有单词的信息。

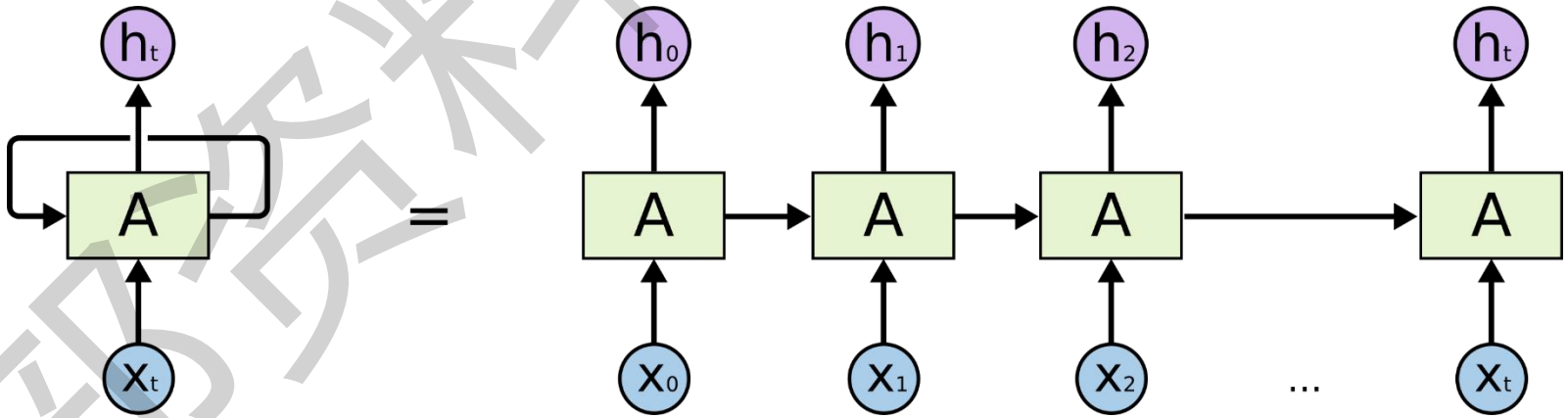


RNN工作-动画演示-5

由于最终输出是从序列的其余部分创建的，因此我们应该能够获取最终输出，并将其传递给前馈层以对意图进行分类

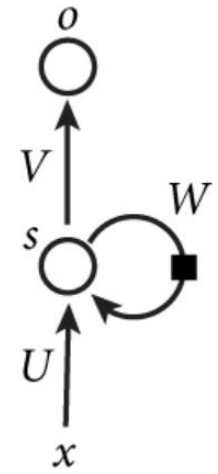


RNN结构

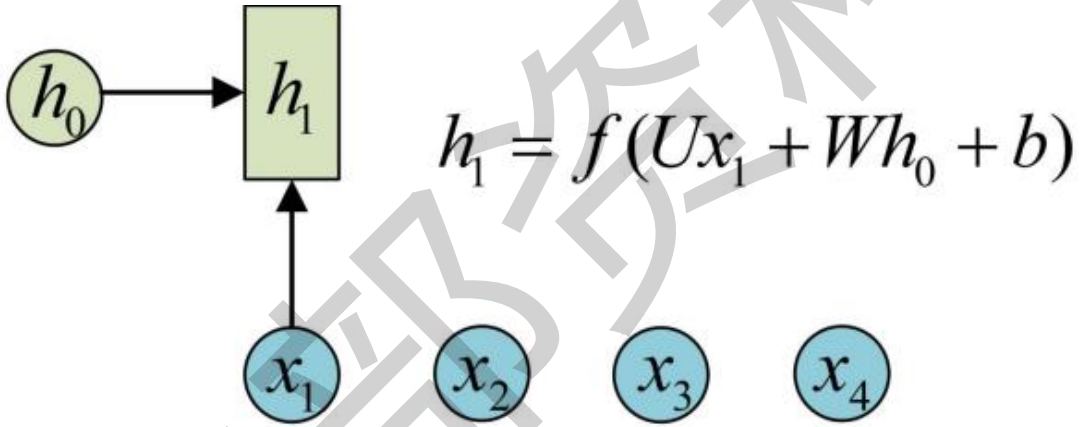


RNN的重要特点

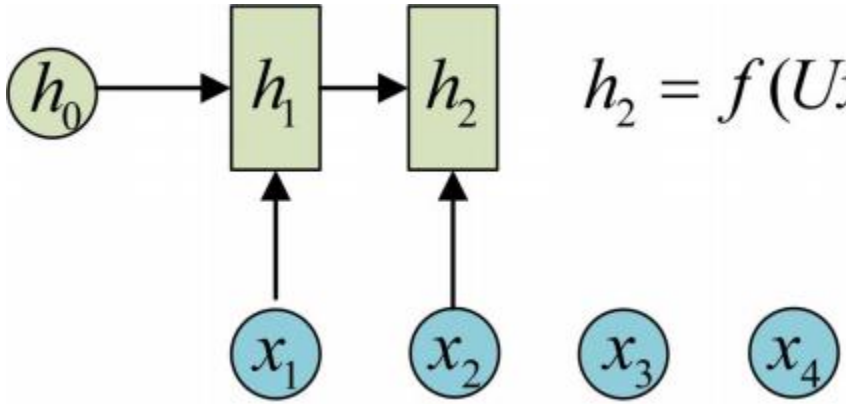
每一步使用的参数U、W、b都是一样的，也就是说每个步骤的参数都是共享的



RNN结构



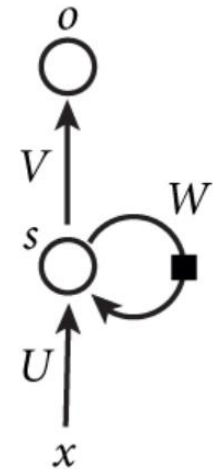
$$h_1 = f(Ux_1 + Wh_0 + b)$$



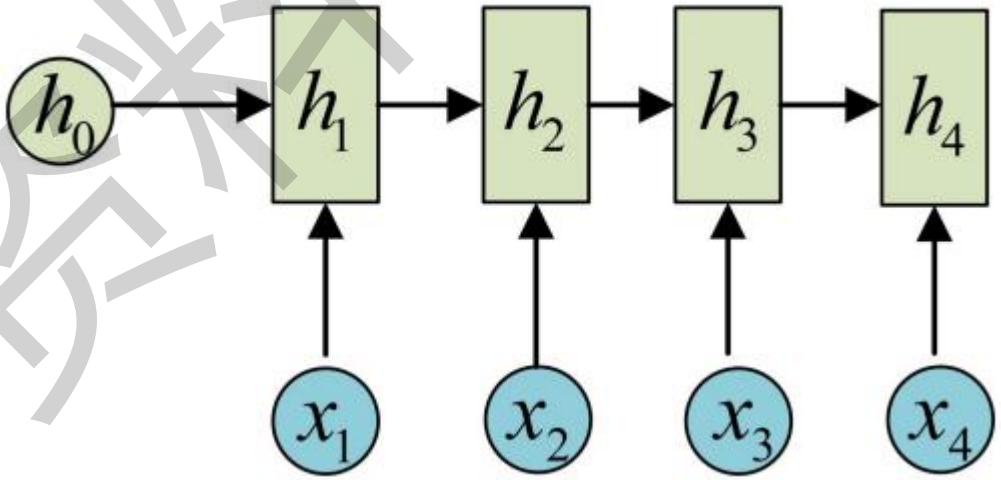
$$h_2 = f(Ux_2 + Wh_1 + b)$$

每次输入的是一个向量，正是由于共享参数，经过反复循环

可以使得RNN可以学到整个序列的信息，输出到最后，就可以表示整个序列的含义

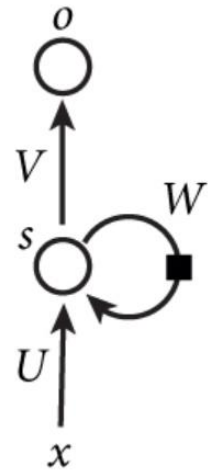


RNN结构

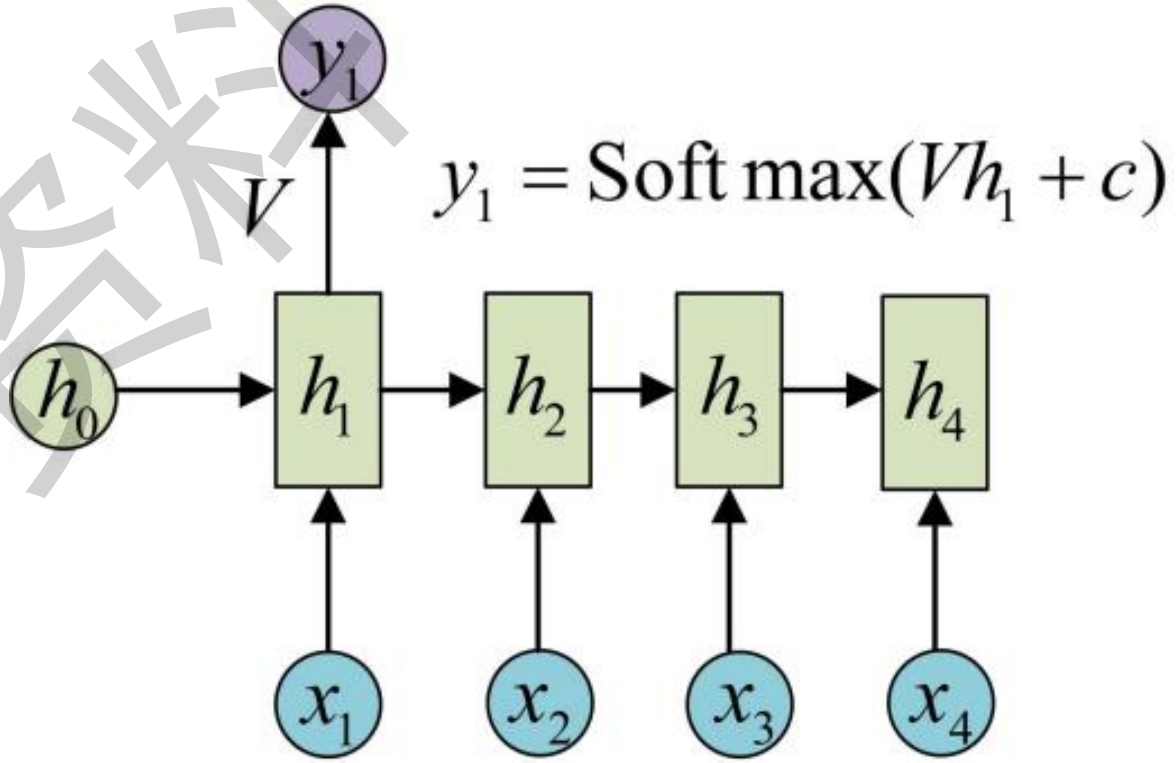


直至最后一个输入结束后，就可以对整个句子进行一个判断

例如，加一个softmax，就可以得到整个句子的分类

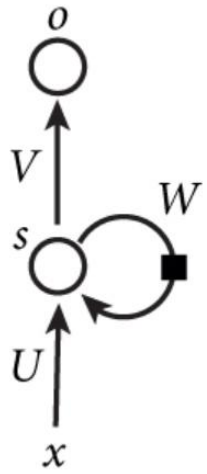


RNN结构



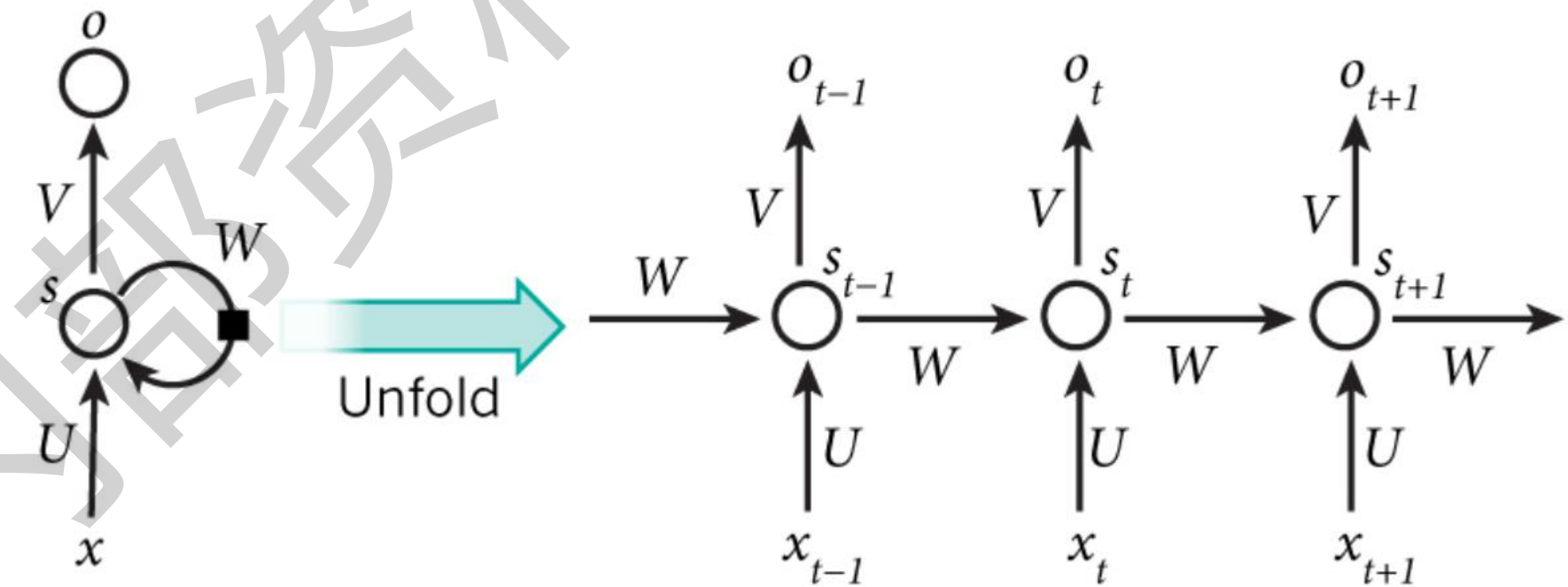
可以对每一个输出进行计算，也可以对最后一个进行判断

例如：视频的每一帧，声音文件的第一个时间节点/时刻

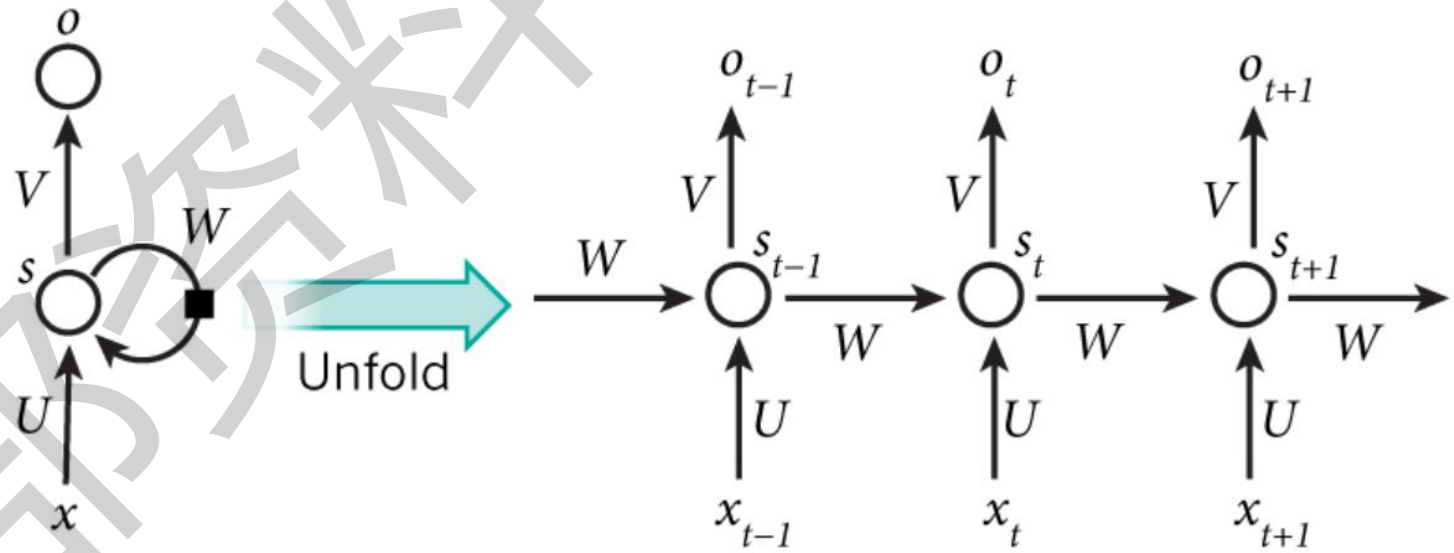


RNN结构

X_t：表示t时刻的输入，o_t：表示t时刻的输出，S_t：表示t时刻的记忆



RNN结构



$$o_t = g(Vs_t) \tag{1}$$

$$s_t = f(Ux_t + Ws_{t-1}) \tag{2}$$

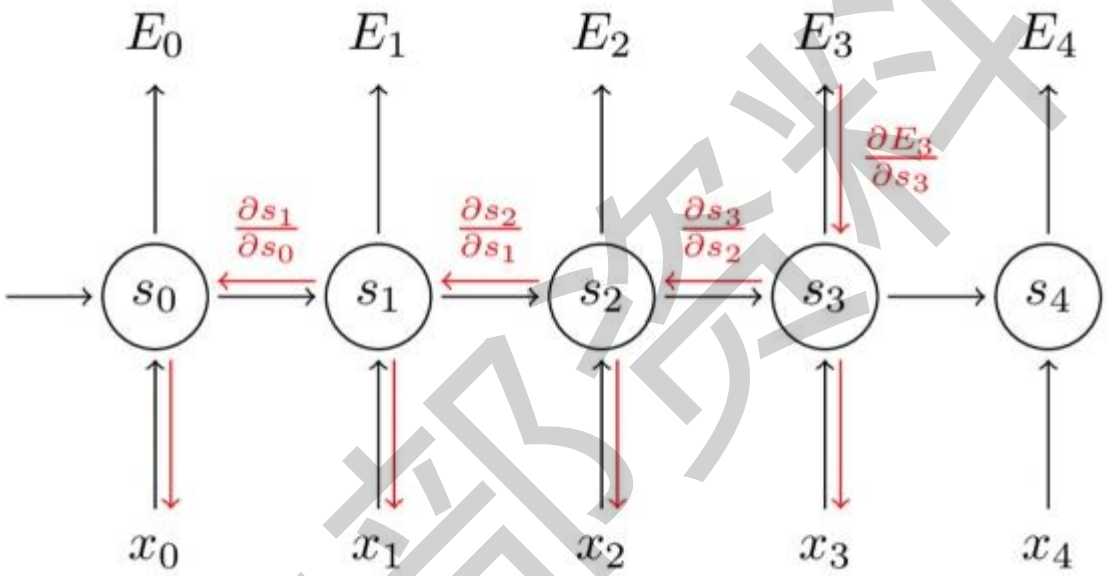
RNN结构

$$o_t = g(Vs_t) \tag{1}$$
$$s_t = f(Ux_t + Ws_{t-1}) \tag{2}$$

如果反复把式2带入到式1，我们将得到：

$$o_t = g(Vs_t) \tag{3}$$
$$= g(Vf(Ux_t + Ws_{t-1}) \tag{4}$$
$$= g(Vf(Ux_t + Wf(Ux_{t-1} + Ws_{t-2})) \tag{5}$$
$$= g(Vf(Ux_t + Wf(Ux_{t-1} + Wf(Ux_{t-2} + Ws_{t-3}))) \tag{6}$$
$$= g(Vf(Ux_t + Wf(Ux_{t-1} + Wf(Ux_{t-2} + Wf(Ux_{t-3} + \dots))))) \tag{7}$$

循环神经网络的训练：BPTT



假设我们对E_3的W求偏导：它的损失首先来源于预测的输出$\hat{y}_3$，
预测的输出又是来源于当前时刻的 记忆s_3，
当前的记忆又是来源于当前的输出和截止到上一时刻的记忆：

$$s_3 = tanh(Ux_3 + Ws_2)$$

因此，根据链式法则可以有:

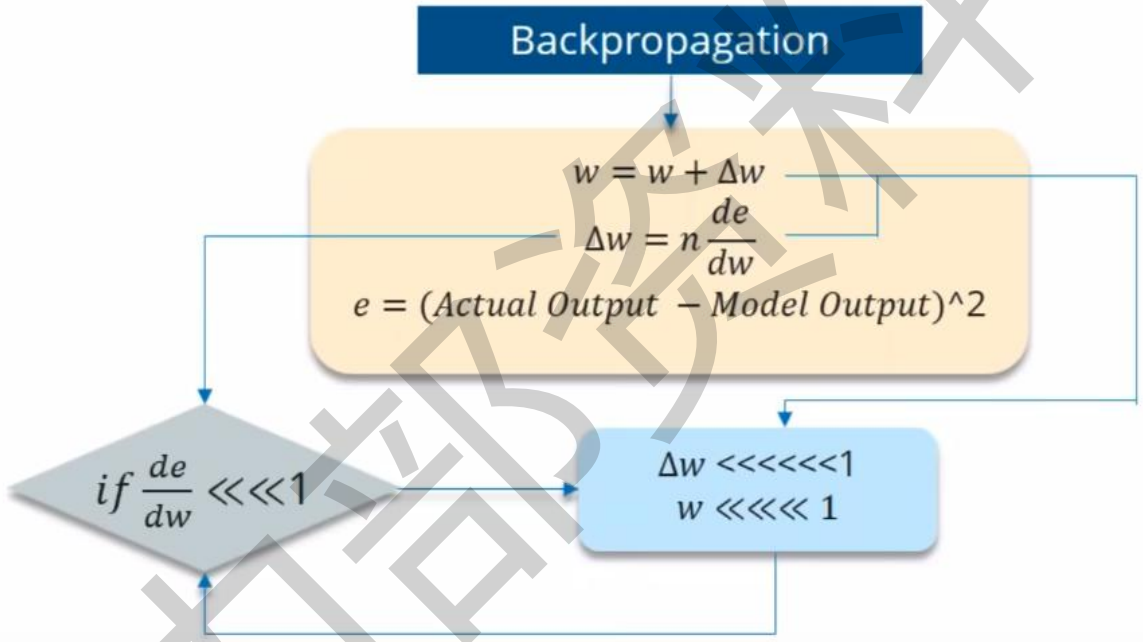
$$\frac{\partial E_3}{\partial W} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial s_3}\frac{\partial s_3}{\partial W}$$

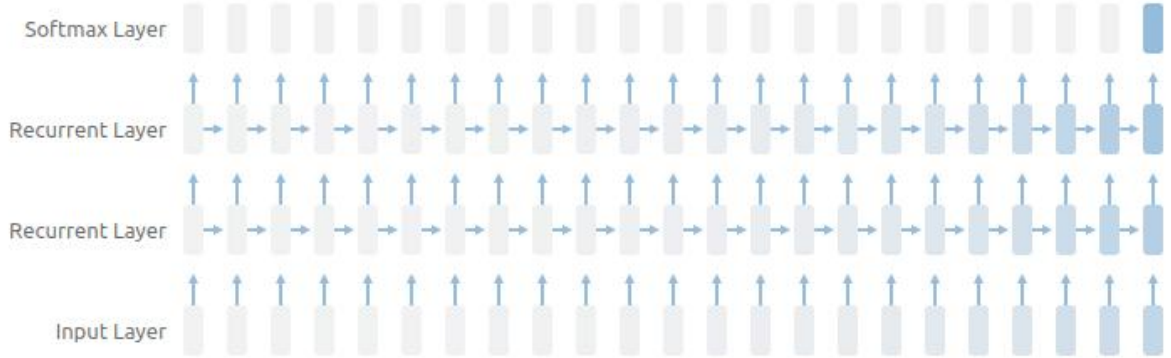
但是，你会发现，$s_2 = tanh(Ux_2 + Ws_1)$，也就是s_2里面的函数还包含了W，
我们不能简单的将 s2视为一个常量，所以真正的链式法则是这样的：

$$\frac{\partial E_3}{\partial W} = \sum_{k=0}^{3} \frac{\partial E_3}{\partial \hat{y_3}}\frac{\partial \hat{y_3}}{\partial s_3}\frac{\partial s_3}{\partial s_k}\frac{\partial s_k}{\partial W}$$

梯度消失

面试必问，解释原因，解决办法



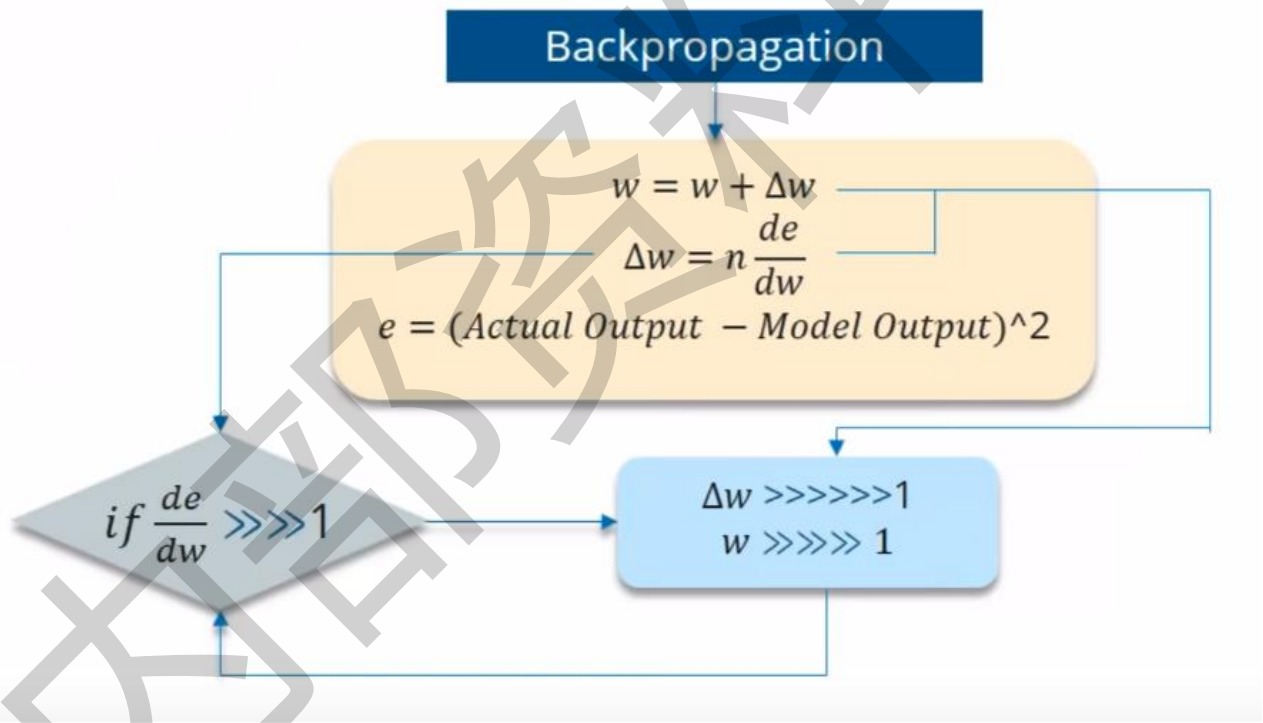


越往前，颜色越来越浅，影响越来越小，参数学习越困难

For ex: 0.863 →0.532 →0.356 →0.192 →0.117 →0.086 →0.023 →0.019..
（一个数不停的去乘以小于1的数字，就会越乘越小，直至梯度消失）

梯度爆炸

面试必问，解释原因，解决办法

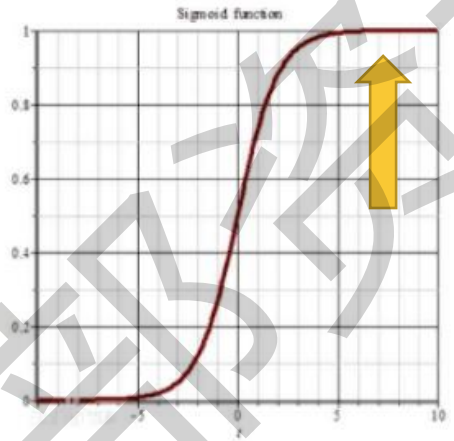


(一个数不停的去乘以大于1的数字，就会越乘越大，直至梯度爆炸)

RNN的梯度消失问题解决思路

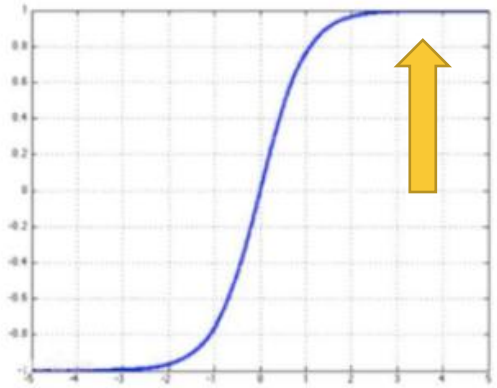
回忆一下，梯度消失问题：

Sigmoid



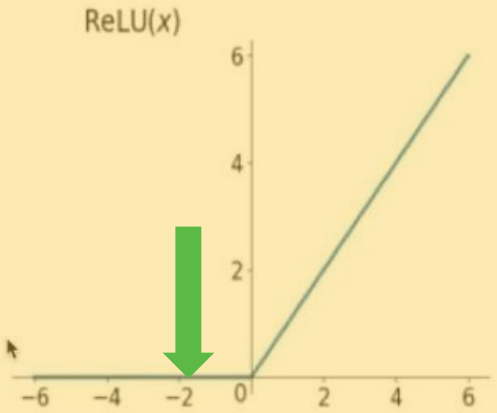
$$f(x)=\frac{1}{1+e^{-x}}$$

tanh



$$f(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$$

ReLU



$$f(x)=\max(0,x)$$

使用relu代替sigmoid和tanh作为激活函数。

使用其他结构的RNN，比如LTSM。

RNN的梯度爆炸问题解决思路

1. 梯度裁剪

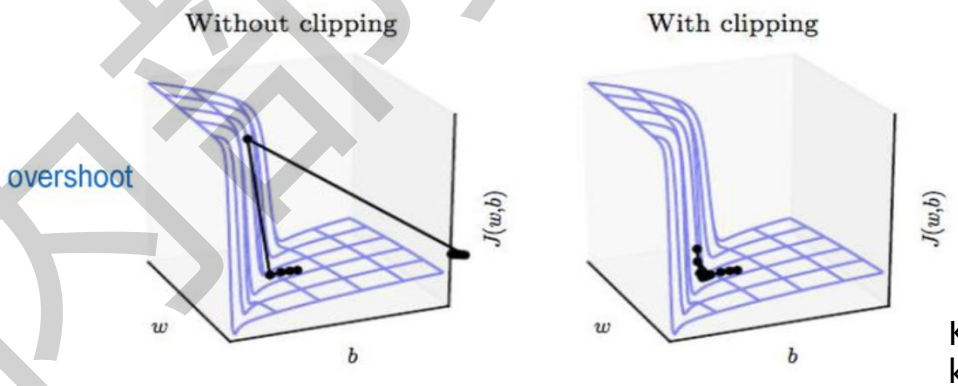
Common solution: clipping gradient

if $||\boldsymbol{g}|| > v$

$$\boldsymbol{g} \leftarrow \frac{\boldsymbol{g}v}{||\boldsymbol{g}||},$$

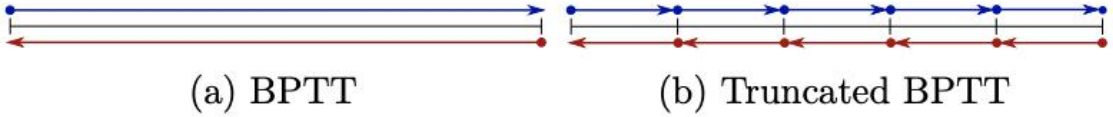
- g: gradient
- v: threshold

If the norm of gradient exceeds some threshold, clip it!



梯度值缩减到阈值之下

2. 截断式的BPTT



(a) BPTT (b) Truncated BPTT

使用两个参数，使得每一段的计算loss并没有那么长，需要谨慎调参

K1：前向传播的时间步。一般来说，这个参数影响模型训练的快慢，即权重更新的频率。
k2：使用BPTT反向传播的时间步。一般来说，参数设置太短，我发学习到语义信息，所以这个参数需要大一点，这样网络能更好的学习序列信息，但是这个参数太大的话可能会导致梯度消失。

RNN的梯度爆炸问题解决思路

其他方法：

使用adam自适应梯度下降等算法，去调节我们的一个学习率，也可以起到比较好的缓解作用

4.RNN的特点

RNN的长依赖问题

There are many animals in the forest， Fish are swimming in the river,

…… ……

The clouds are in the _____.

亲！我们需要往前搜集许多词，再来做推断填什么吗？

John出生在英国，并且在那里长大，他的家乡靠近海边，风景很漂亮，他喜欢去海边玩……

……

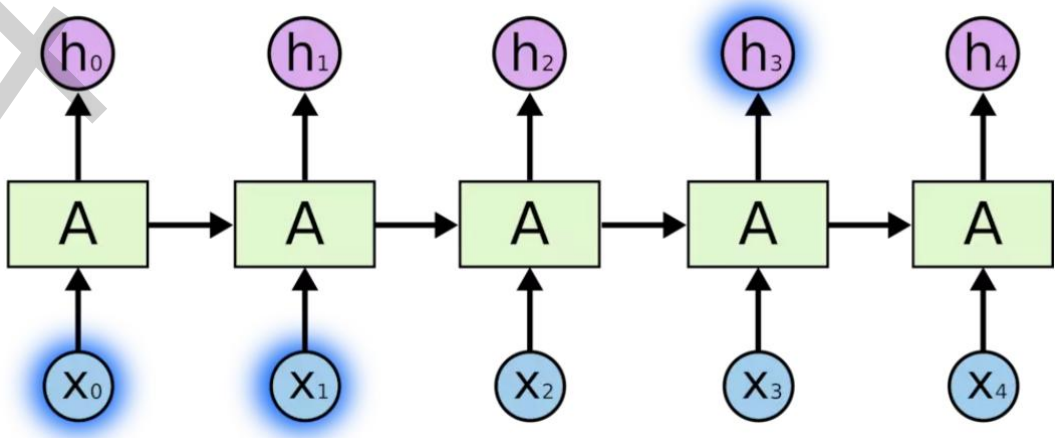
所以，John说_____语。

要推断这个填什么，就需要往前搜集需要词了。

RNN的长依赖问题

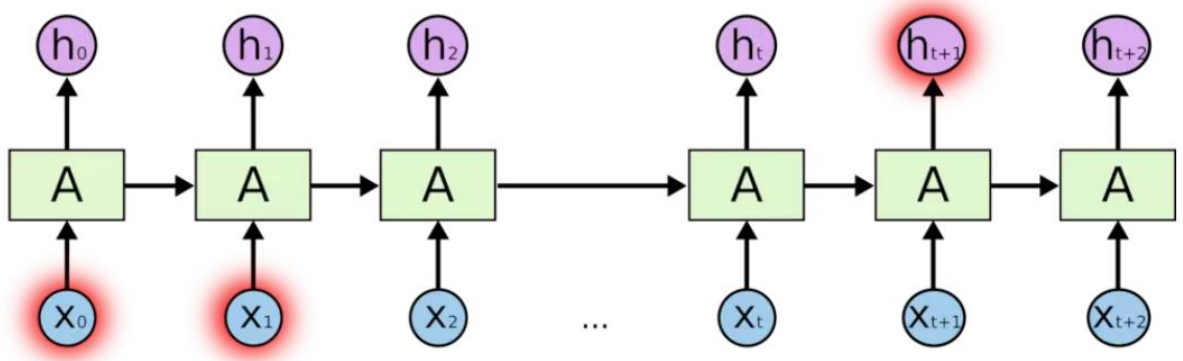
位置间隔不太长时

RNN模型很容易学习到这些信息



短依赖问题

位置间隔很长时



长依赖问题

RNN 就很难学习到这些离得很远的信息

优点

- 与人工神经网络相比，RNN的主要优点是RNN可以处理序列数据进行；
- 参数共享。

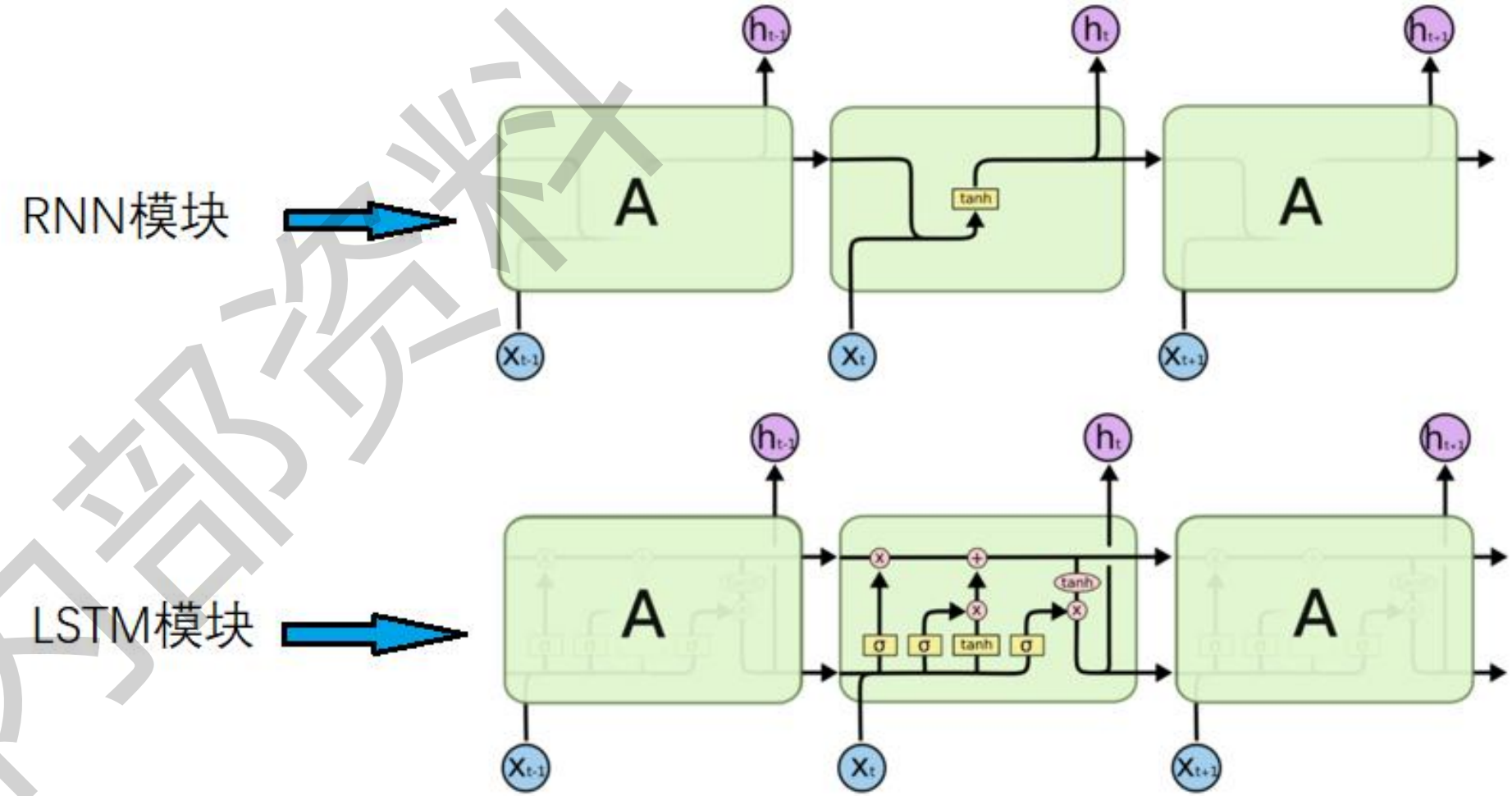
缺点

- 梯度下降和梯度消失；
- 训练RNN网络非常困难；
- 无法处理长序列问题。

5.LSTM结构原理

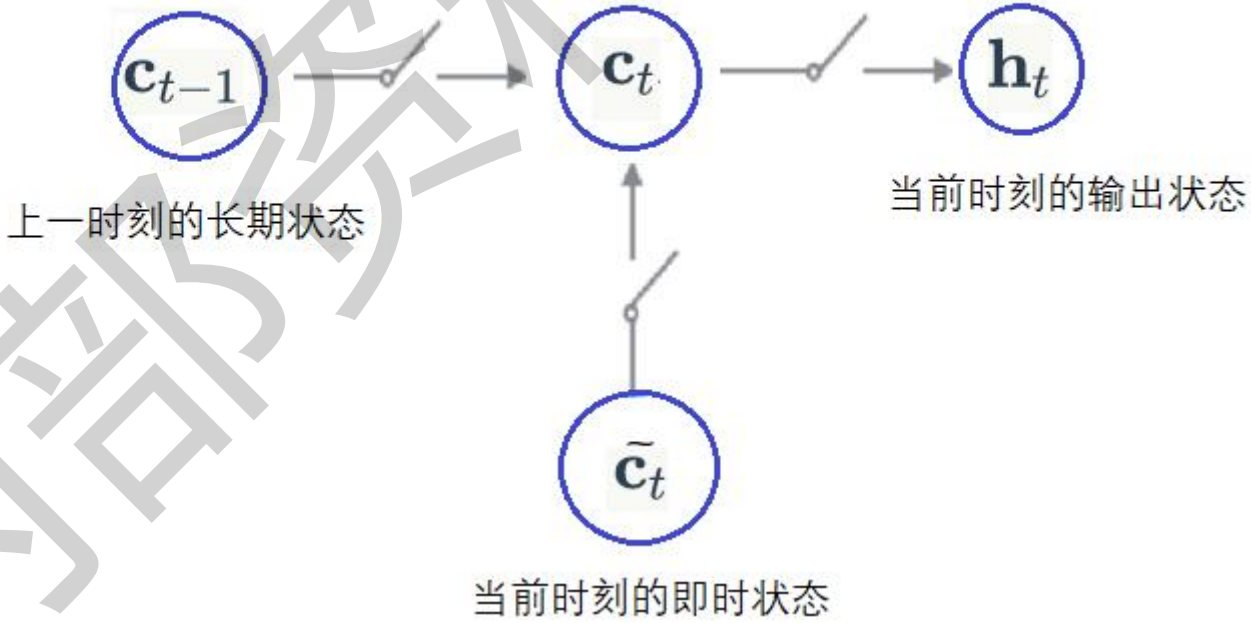
Long Short Term Memory（LSTM）

LSTM与RNN的结构比较



LSTM的门结构

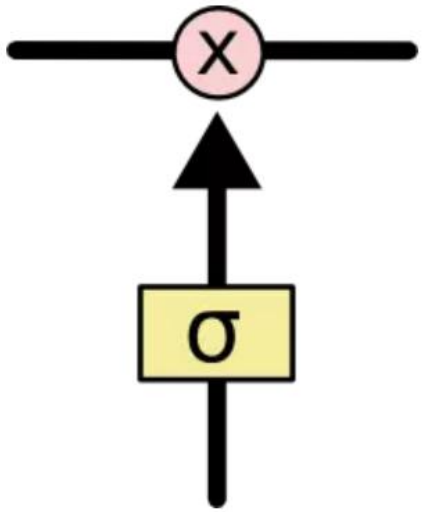
长期状态c的控制：



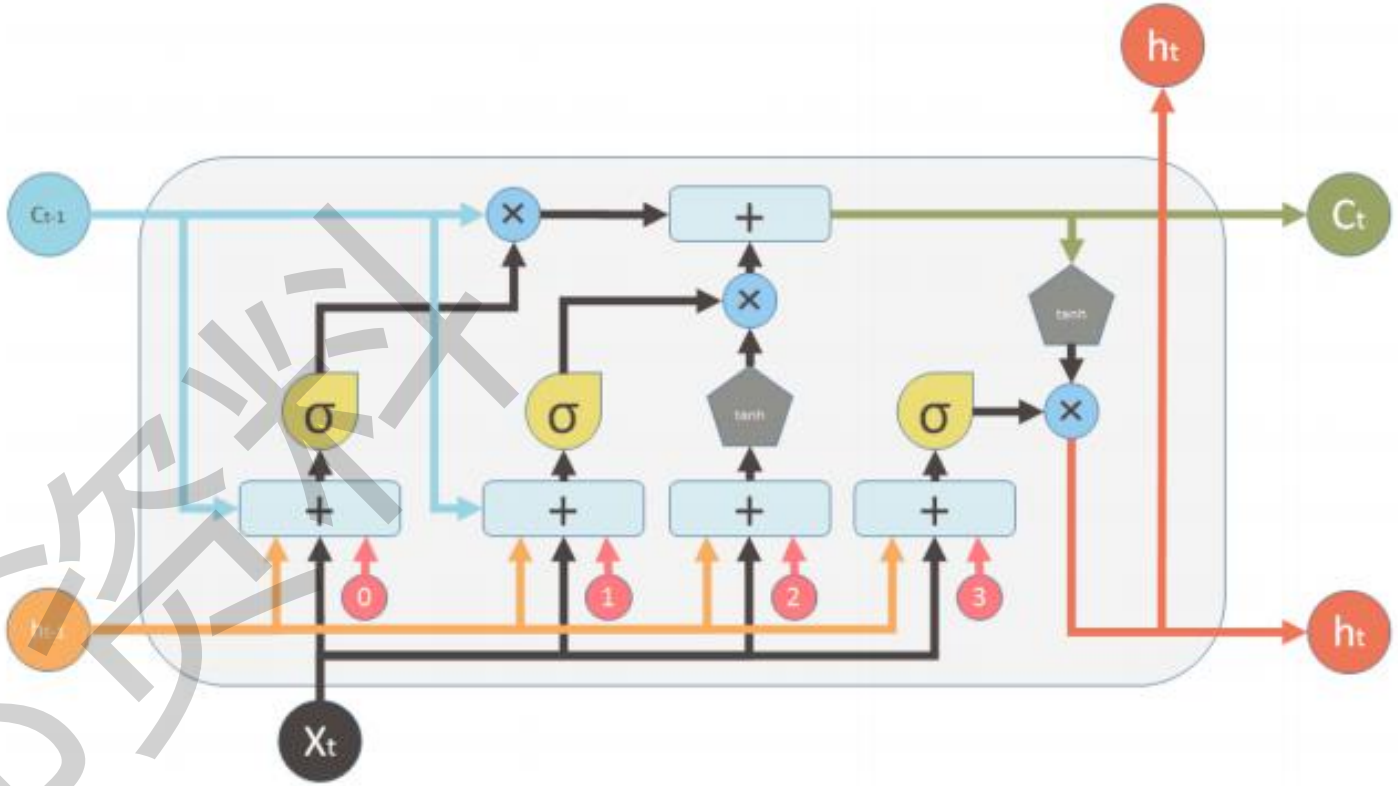
上一时刻的长期状态

当前时刻的输出状态

当前时刻的即时状态



门结构



Inputs:

- X_t Input vector
- C_{t-1} Memory from previous block
- h_{t-1} Output of previous block

outputs:

- C_t Memory from current block
- h_t Output of current block

Nonlinearities:

- σ Sigmoid
- tanh Hyperbolic tangent

Bias: 0

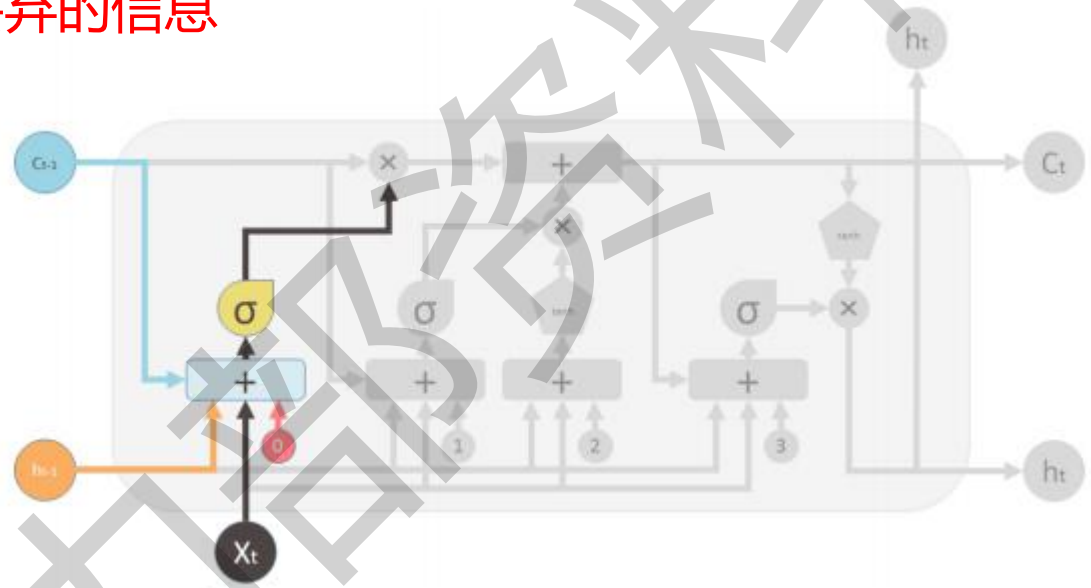
Vector operations:

- × Element-wise multiplication
- + Element-wise Summation / Concatenation

LSTM的工作过程

第一步是决定我们丢弃什么信息（遗忘门forget gate）

丢弃的信息



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

对比：$f_t = \sigma(W\mathbf{x} + \mathbf{b})$

原计划今天是John来给大家上课，但是他临时有事来不了，

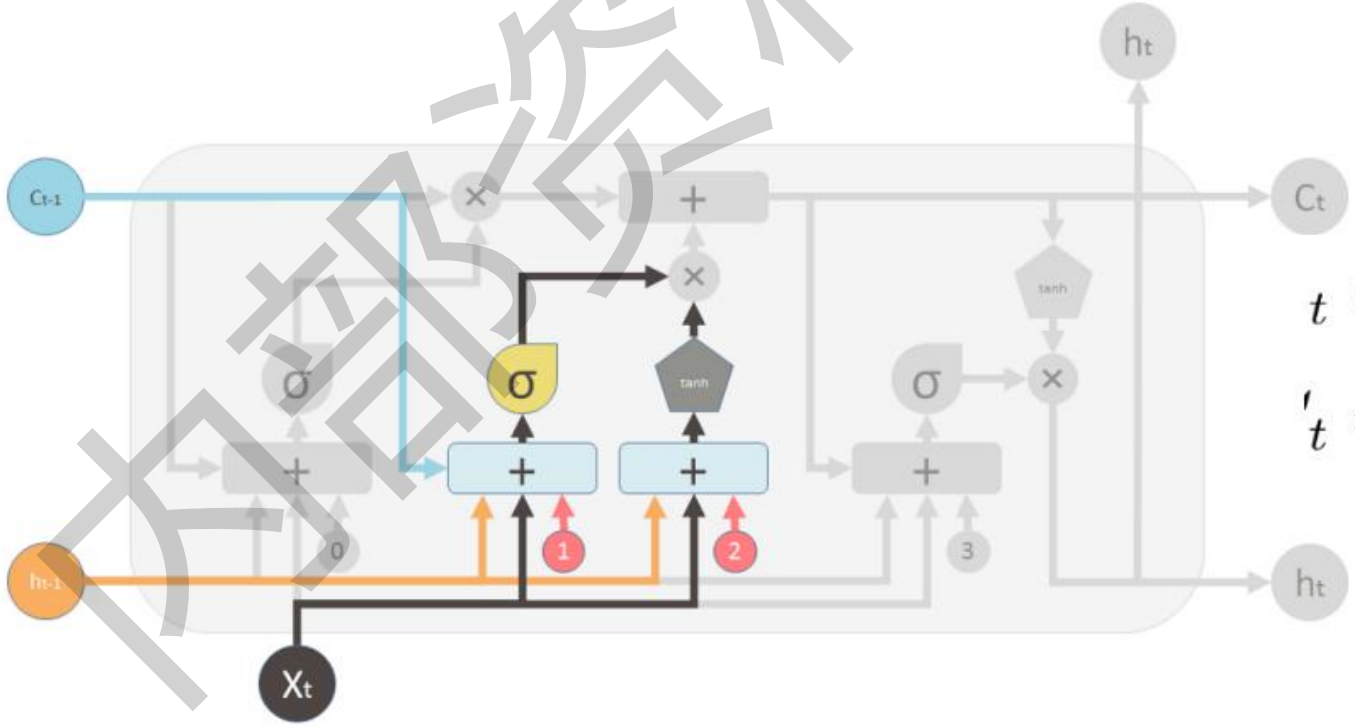
所以今天的课程由我来给大家上，接下来，_____要给大家讲的内容是……

那么从这里往后就要舍弃掉 “John” 的相关信息，只关注 “我” 的信息

LSTM的工作过程

第二步是确定什么样的新信息被存放在单元状态中（输入门input / upgate gate）

要输入的新信息



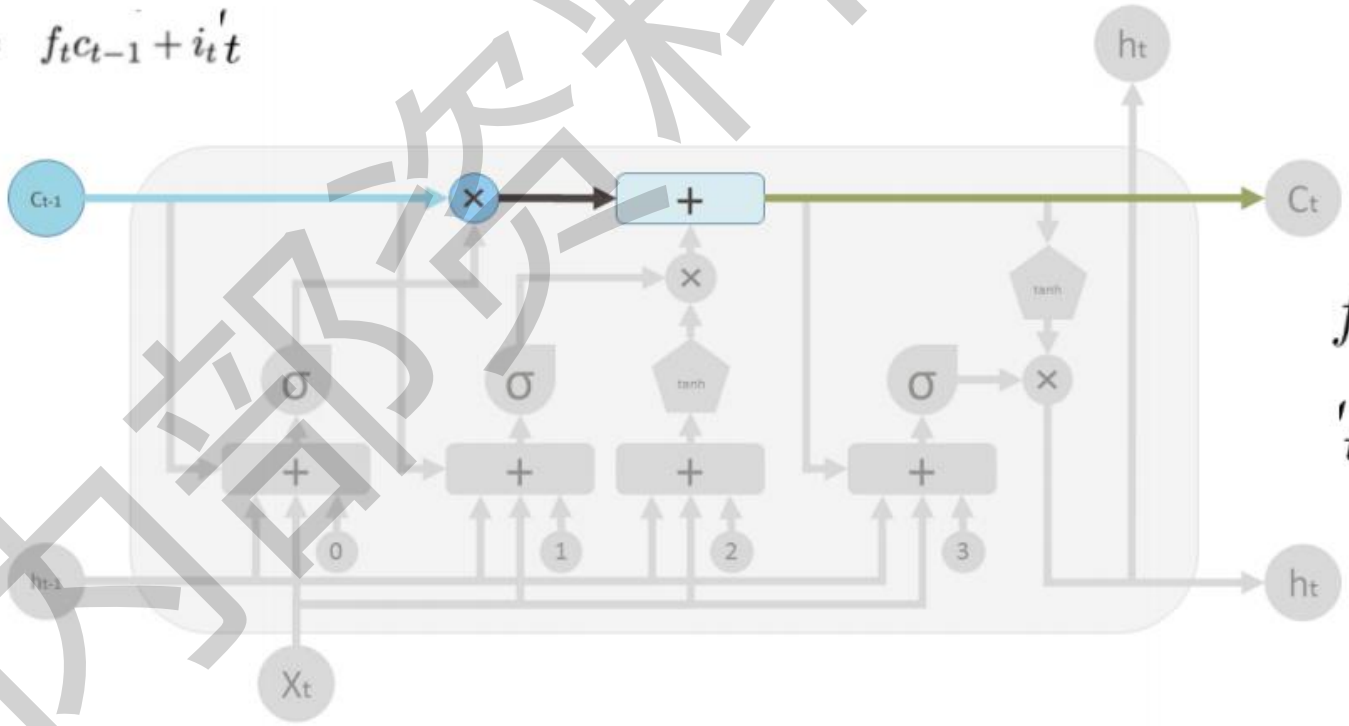
$$t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$'_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

LSTM的门结构

记忆单元

$$c_t = f_t c_{t-1} + i_t' t$$

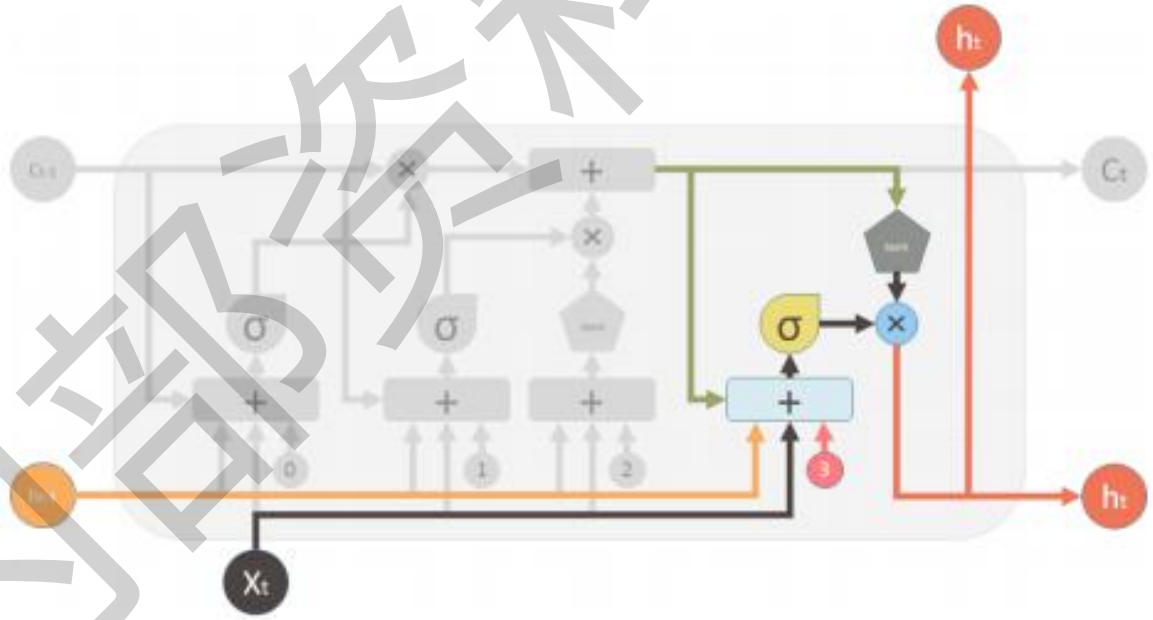


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

$$'_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

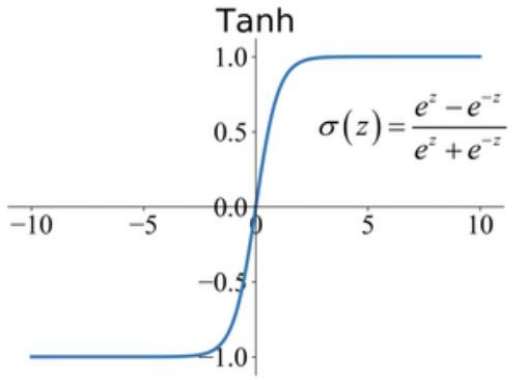
LSTM的工作过程

输出门



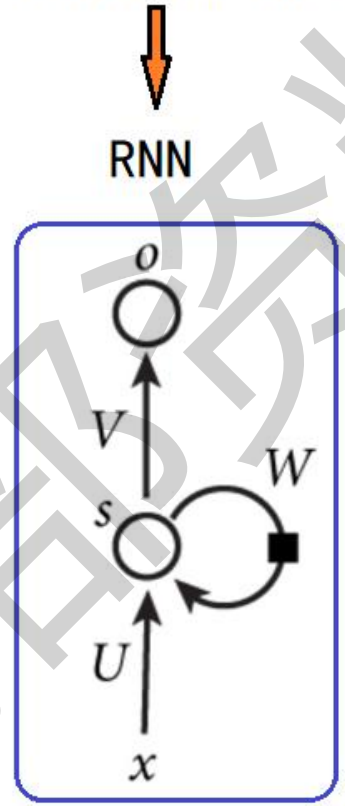
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

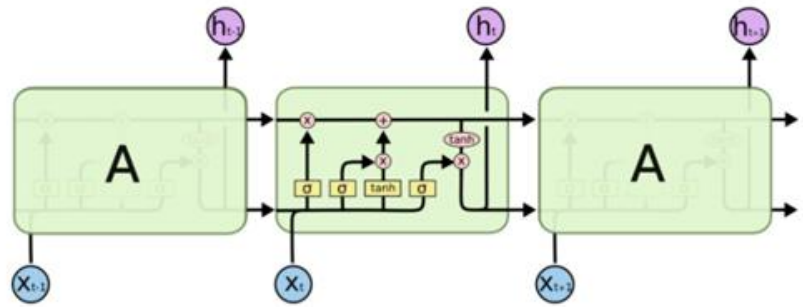
$$h_t = o_t * \tanh\left(C_t\right)$$



小结

输入是元素间不独立的序列





LSTM

RNN无法处理长时依赖问题

7.双向循环神经网络

双向循环神经网络

对于语言模型来说，很多时候光看前面的词是不够的，比如下面这些话：

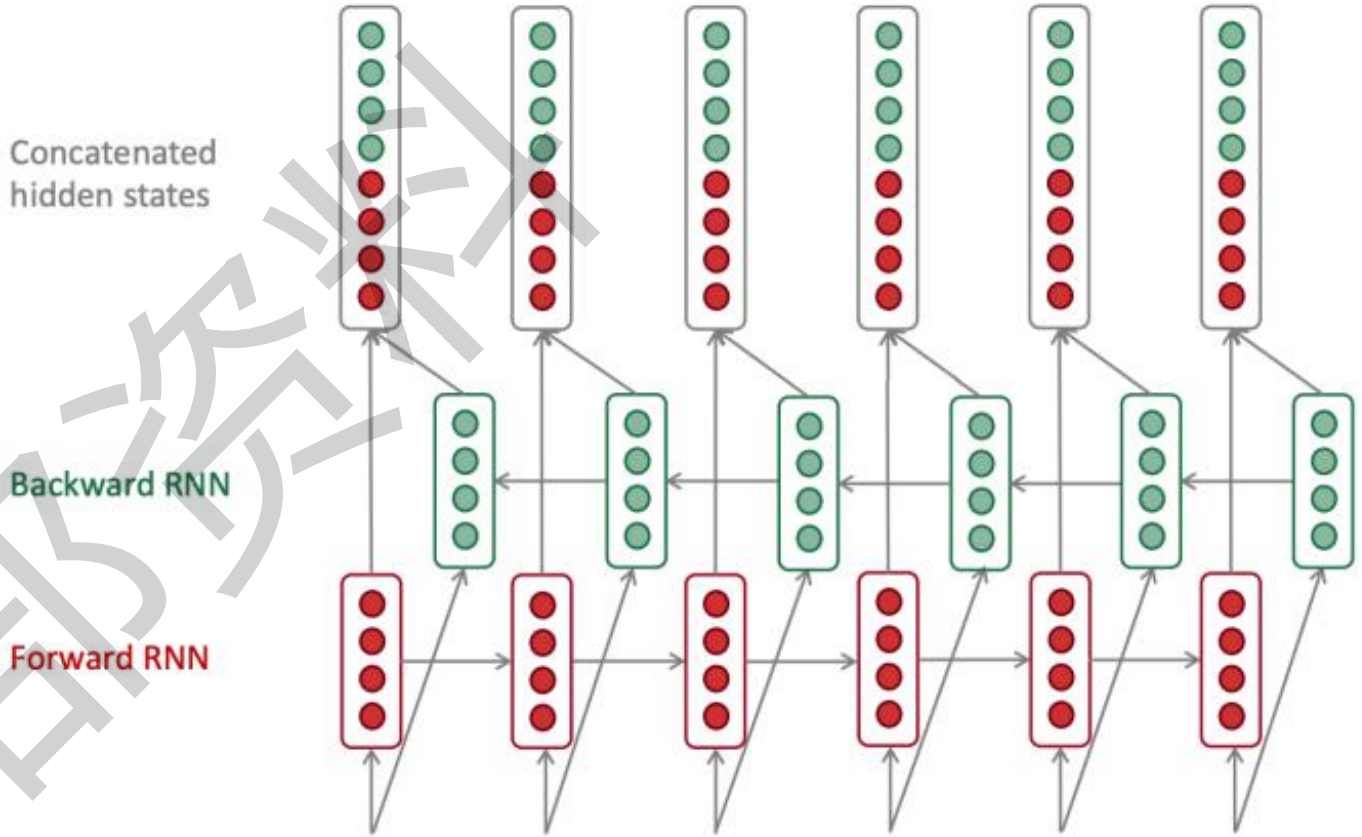
我的水杯摔碎了，我要“____” 

亲！你猜我要做什么？

我的水杯摔碎了，我要“____”一个新的水杯。

你不会猜错了吧？

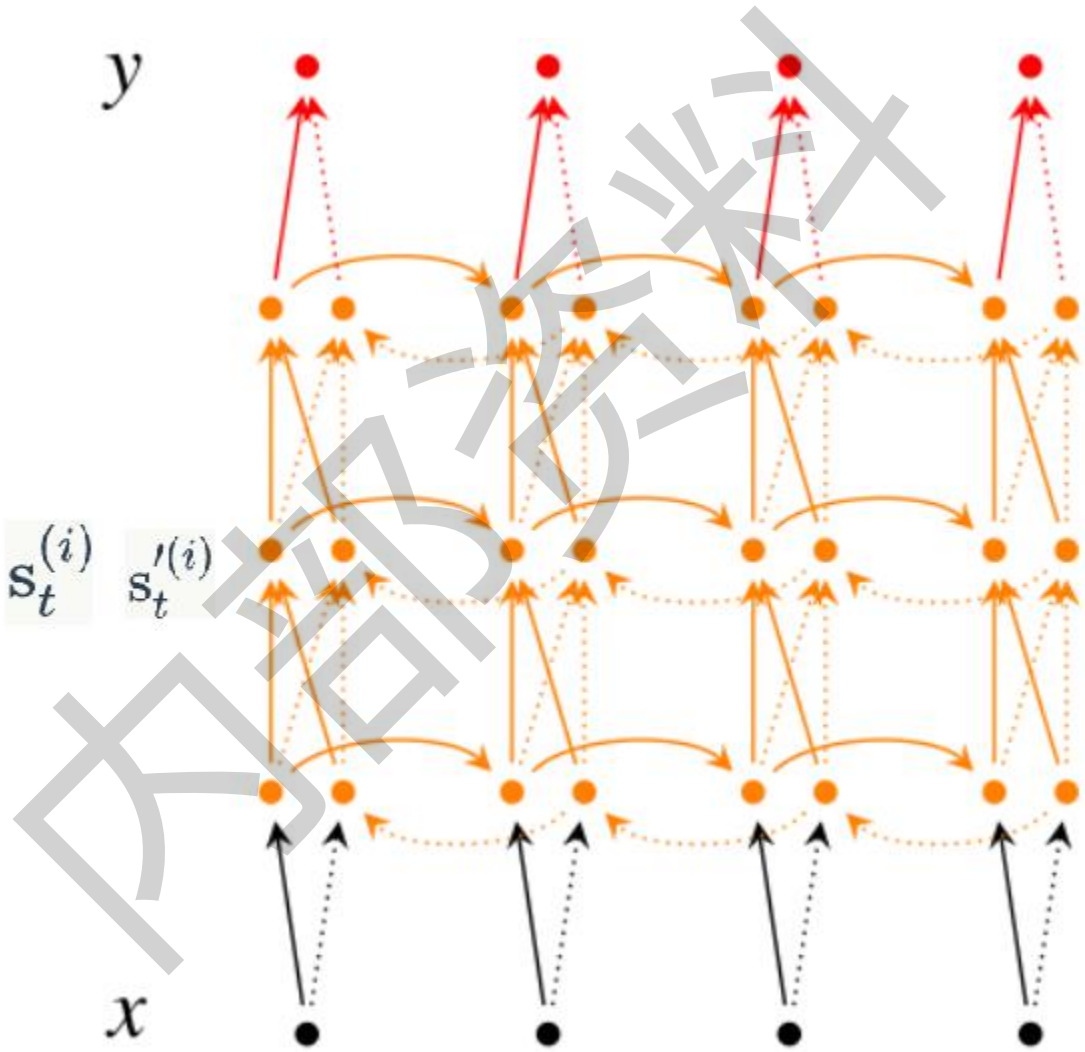
双向循环神经网络



双向，两组RNN，并行，两组信息进行拼接

从右到左（希望在前文中就可以捕获到下文的信息），单向无法解决！

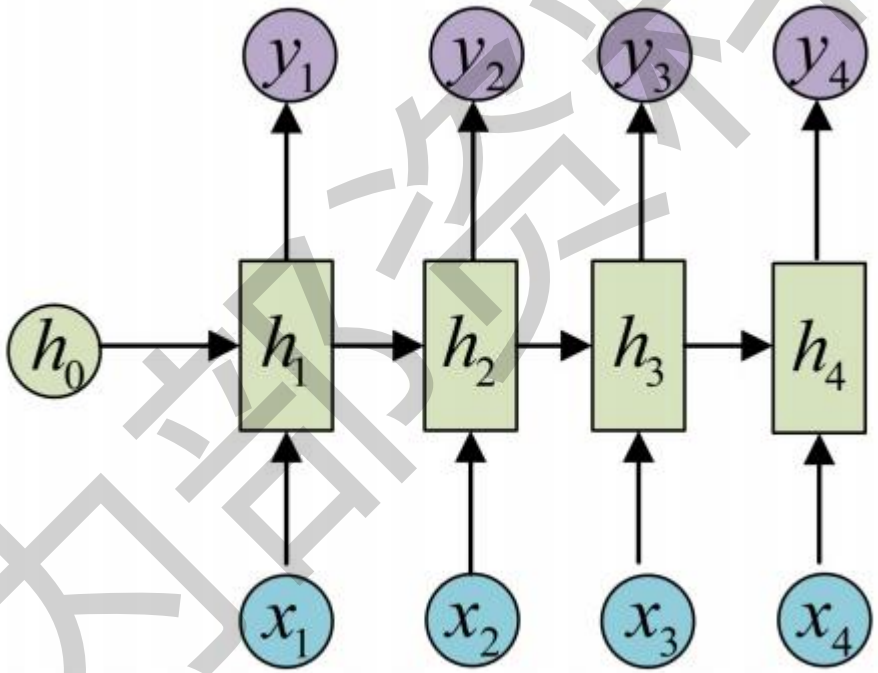
双向循环神经网络



$$
\begin{aligned}
o_t &= g(V^{(i)} s_t^{(i)} + V'^{(i)} s_t'^{(i)}) \\
s_t^{(i)} &= f(U^{(i)} s_t^{(i-1)} + W^{(i)} s_{t-1}) \\
s_t'^{(i)} &= f(U'^{(i)} s_t'^{(i-1)} + W'^{(i)} s'_{t+1}) \\
&\cdots \\
s_t^{(1)} &= f(U^{(1)} x_t + W^{(1)} s_{t-1}) \\
s_t'^{(1)} &= f(U'^{(1)} x_t + W'^{(1)} s'_{t+1})
\end{aligned}
$$

8.RNN/LSTM的常用结构

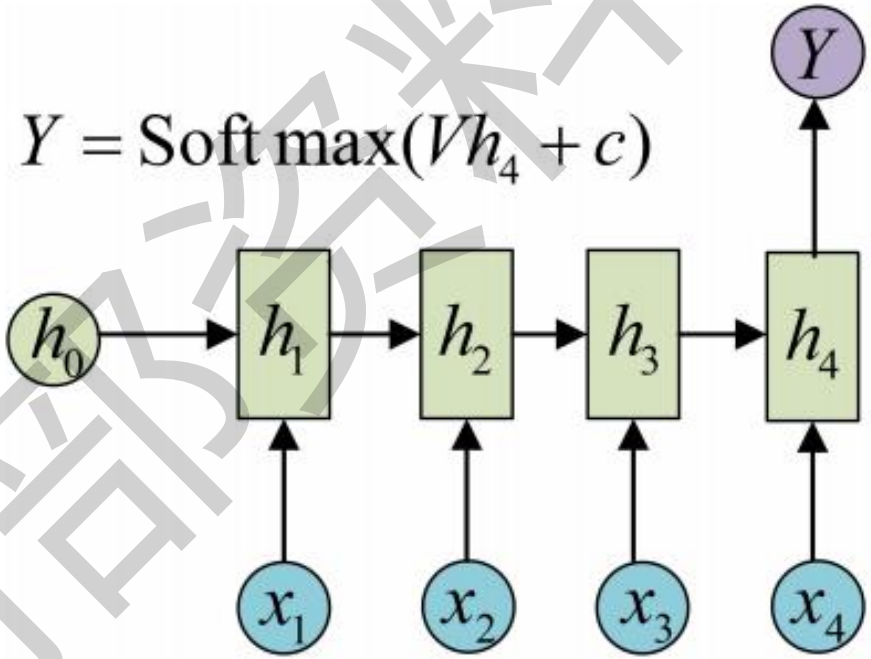
N vs N



命名实体识别
计算视频中每一帧的分类标签
因为要对每一帧进行计算，因此输入和输出序列等长

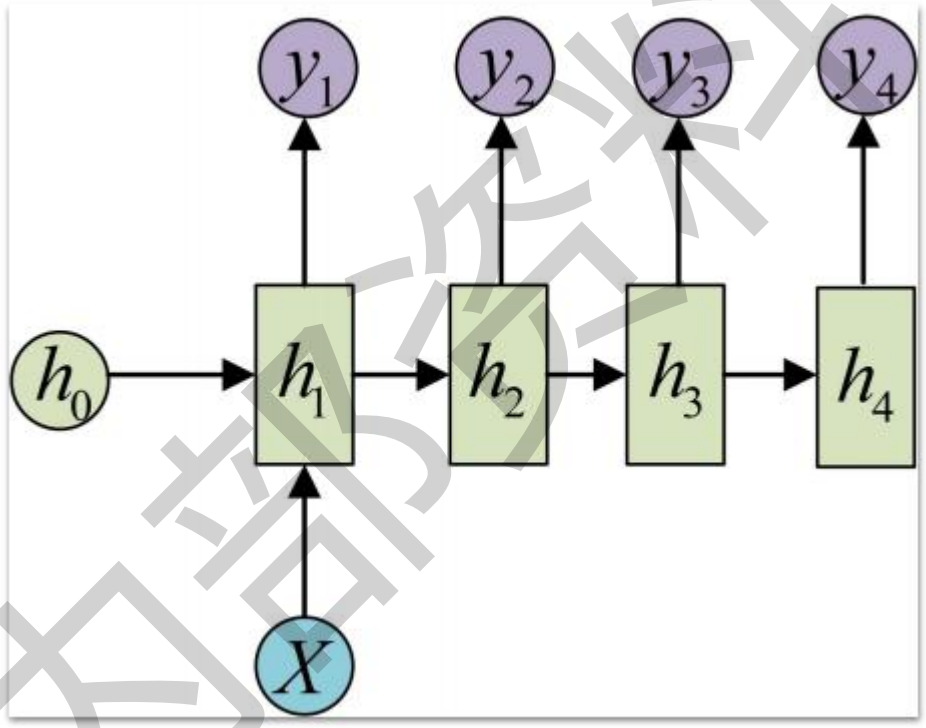
应用的并不是特别多，仅以上

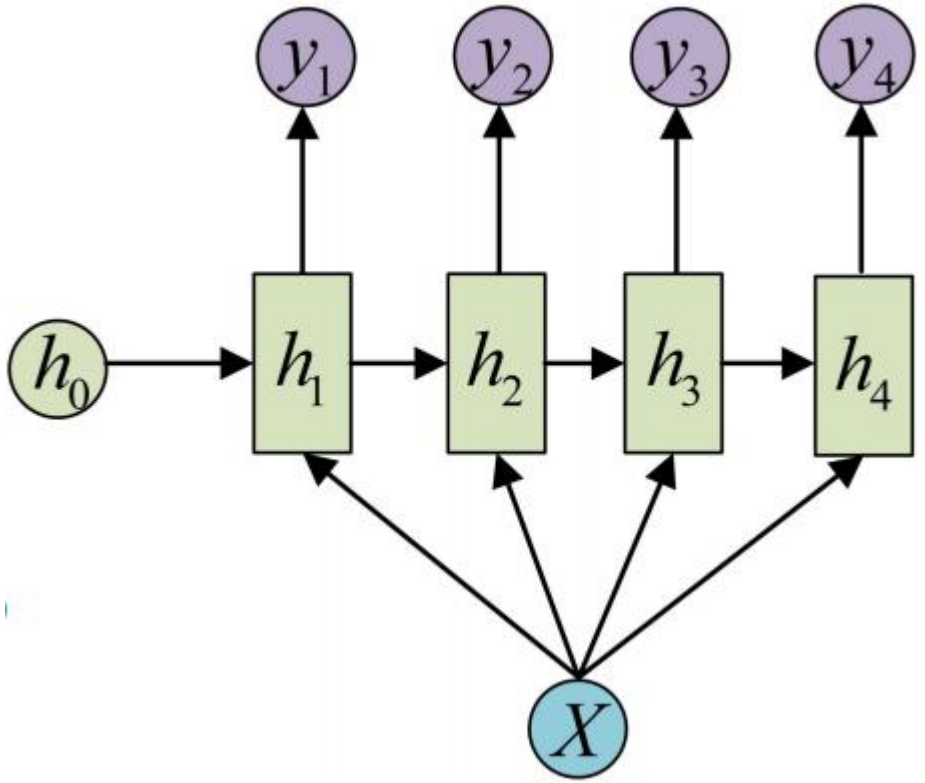
N vs 1



文本分类、情感分析

1 vs N

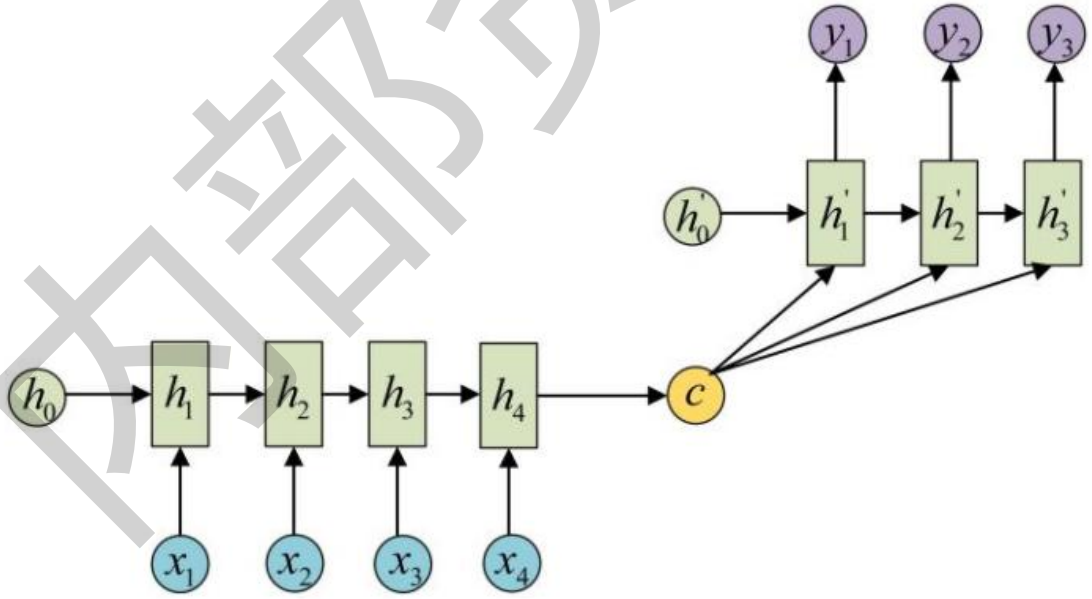


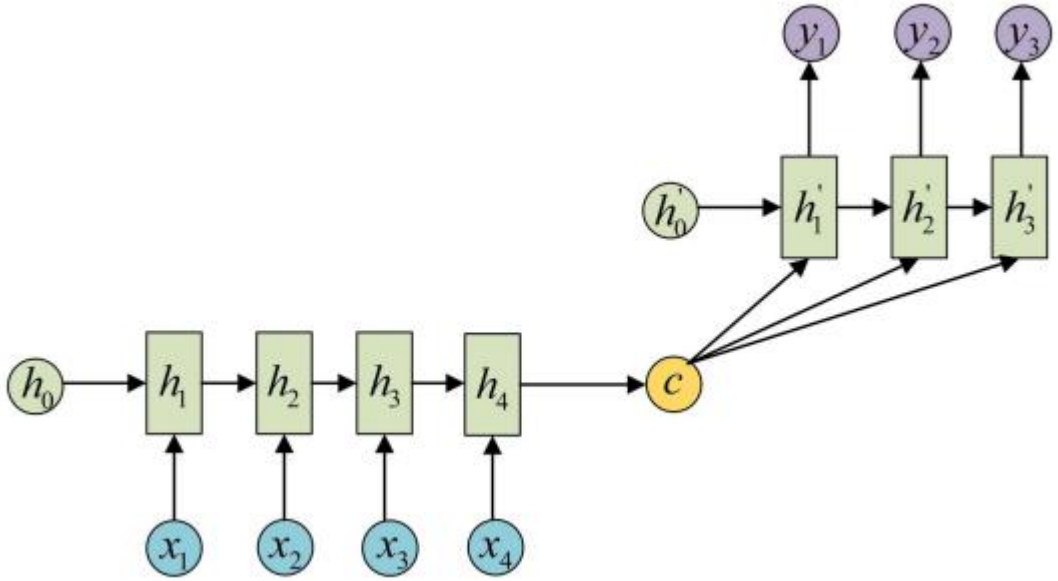


从图像生成文字（image caption），此时输入的X就是图像的特征，而输出的y序列就是一段句子；
从类别（单个)生成语音或音乐（音符/谱子）等
两种都可以，可以对比尝试哪种更好一些！

N vs M（Seq2Seq模型）

- 机器翻译。Encoder-Decoder的最经典应用，事实上这一结构就是在机器翻译领域最先提出的。
- 文本摘要。输入是一段文本序列，输出是这段文本序列的摘要序列。
- 阅读理解。将输入的文章和问题分别编码，再对其进行解码得到问题的答案。
- 语音识别。输入是语音信号序列，输出是文字序列。
- ……

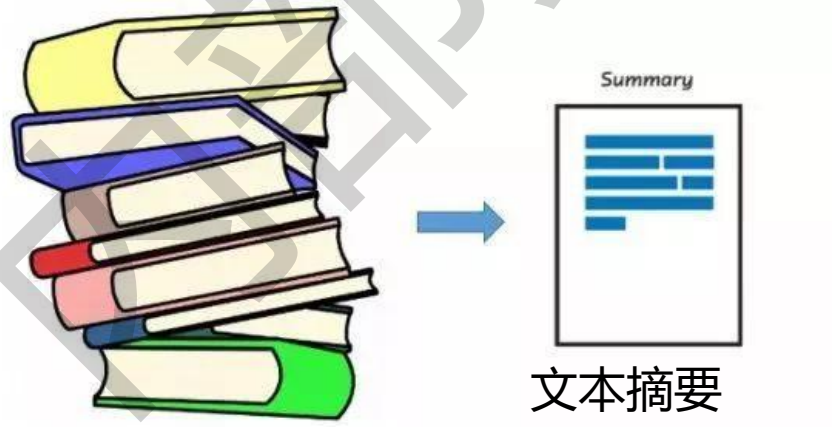




N vs M 应用

文本翻译

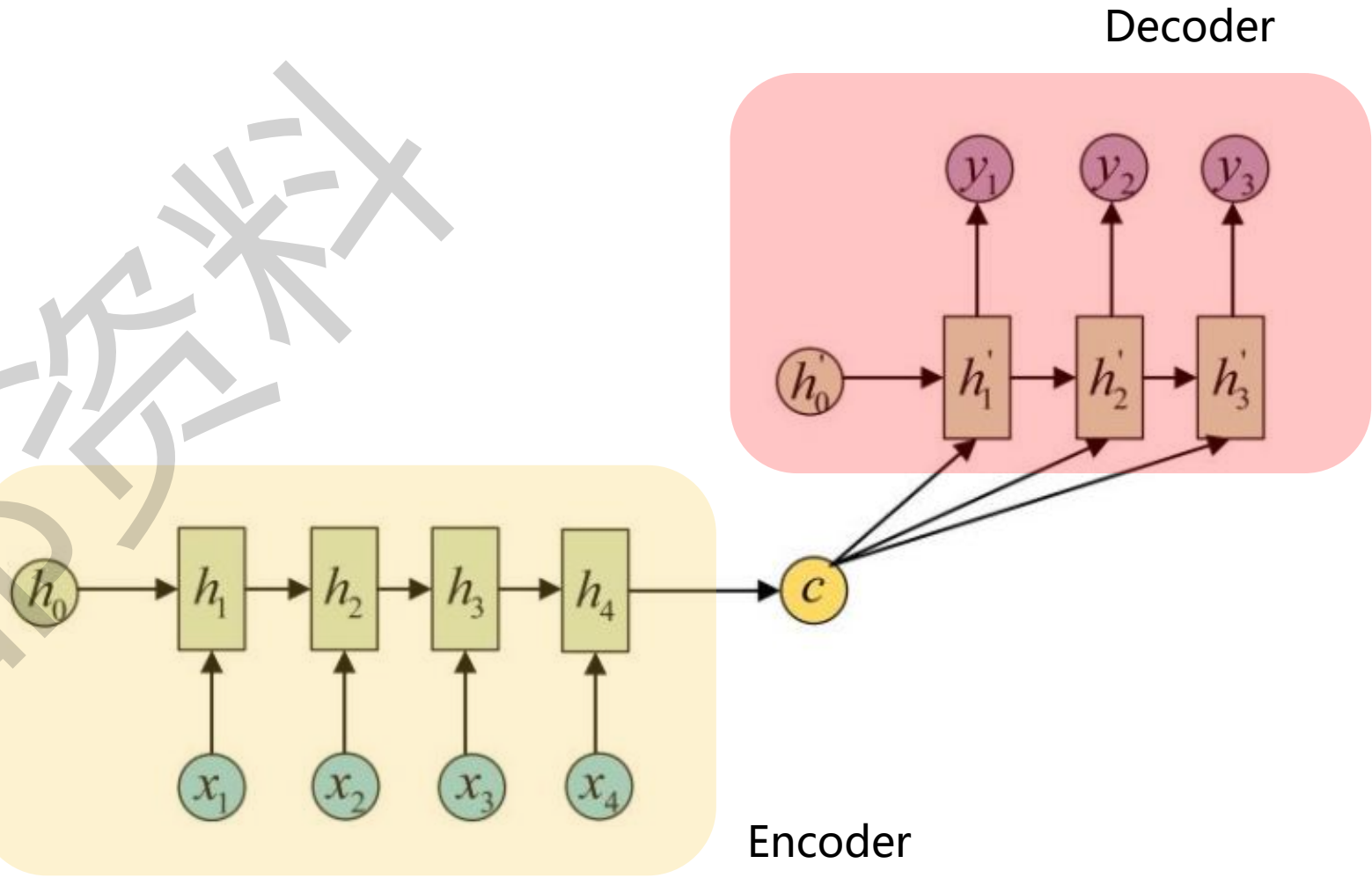
语音转文本



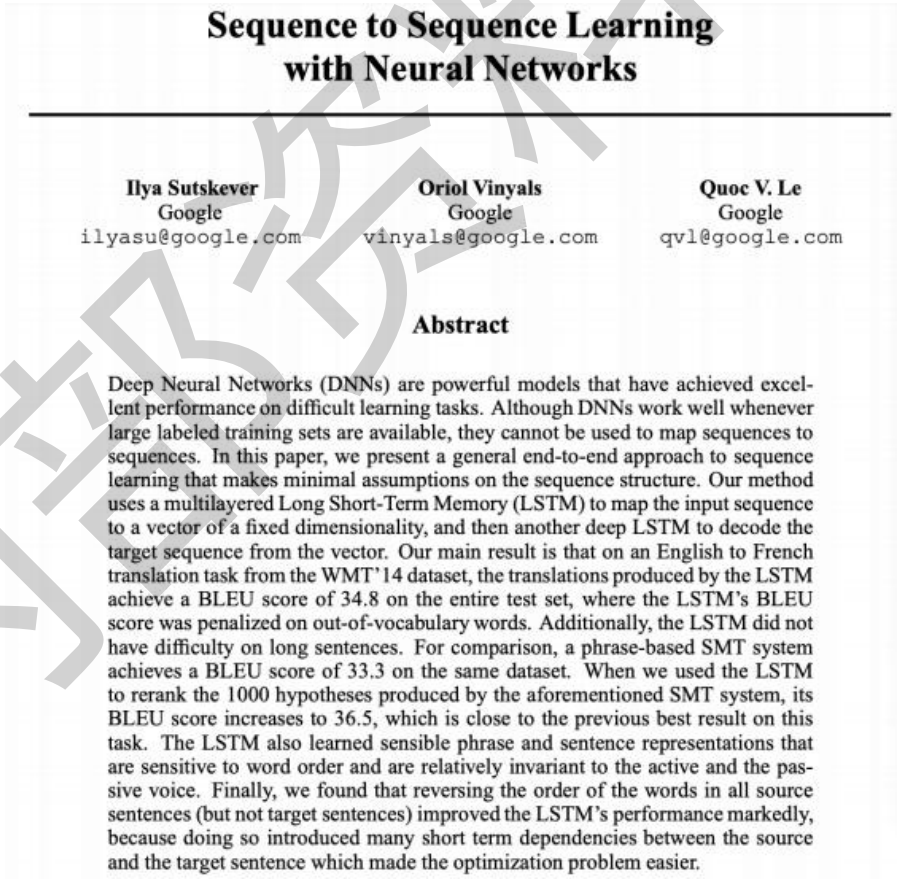
文本摘要

视频摘要

N vs M (Seq2Seq模型)



Definition of the Sequence to Sequence Model



Sequence to Sequence Learning with Neural Networks

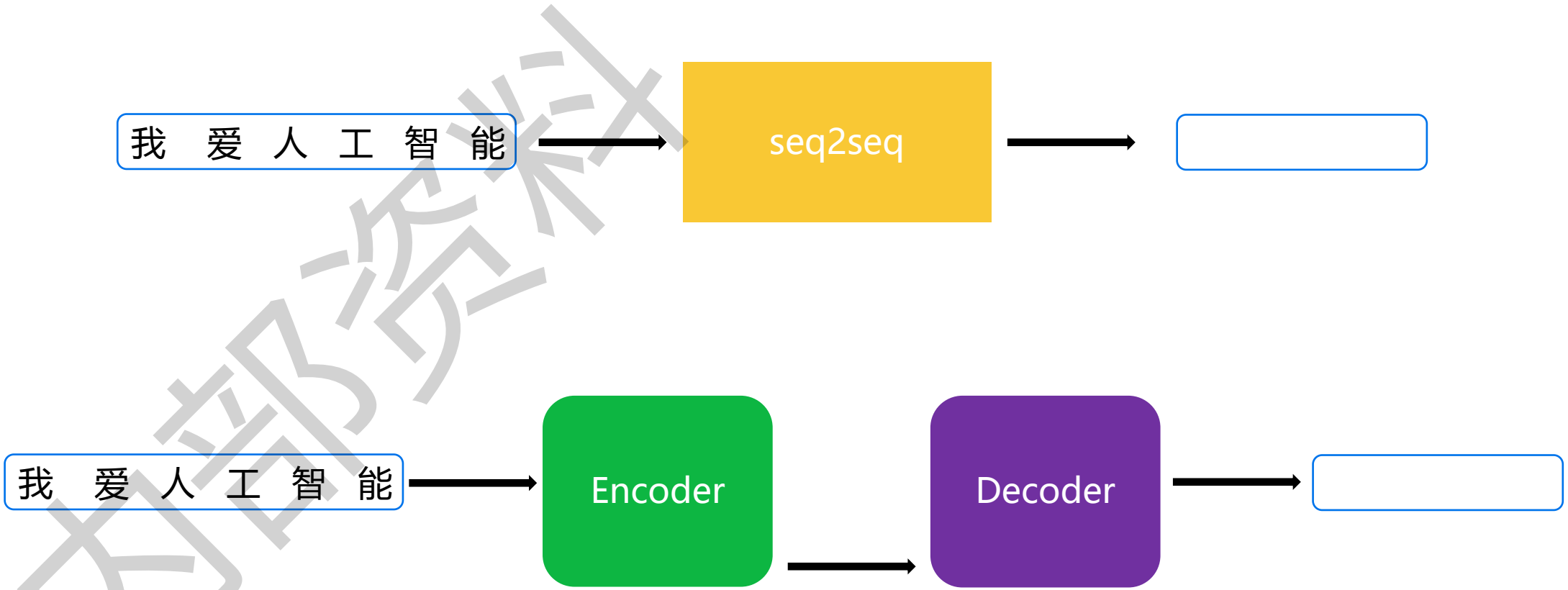
Ilya Sutskever
Google
ilyasu@google.com

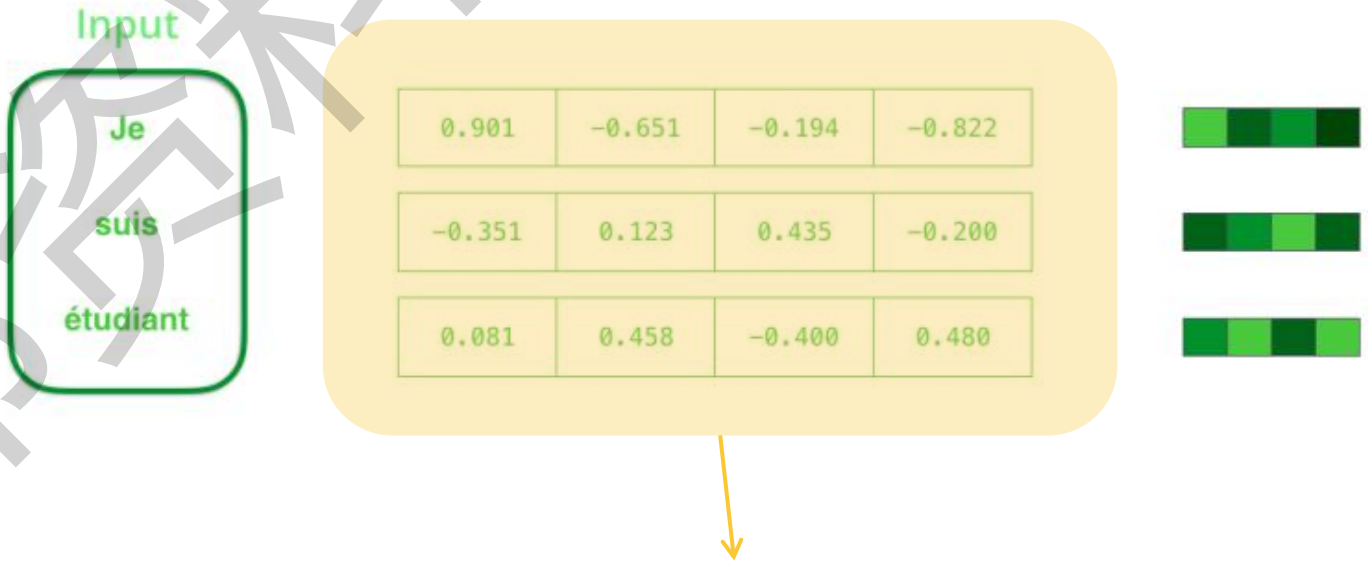
Oriol Vinyals
Google
vinyals@google.com

Quoc V. Le
Google
qvl@google.com

Abstract

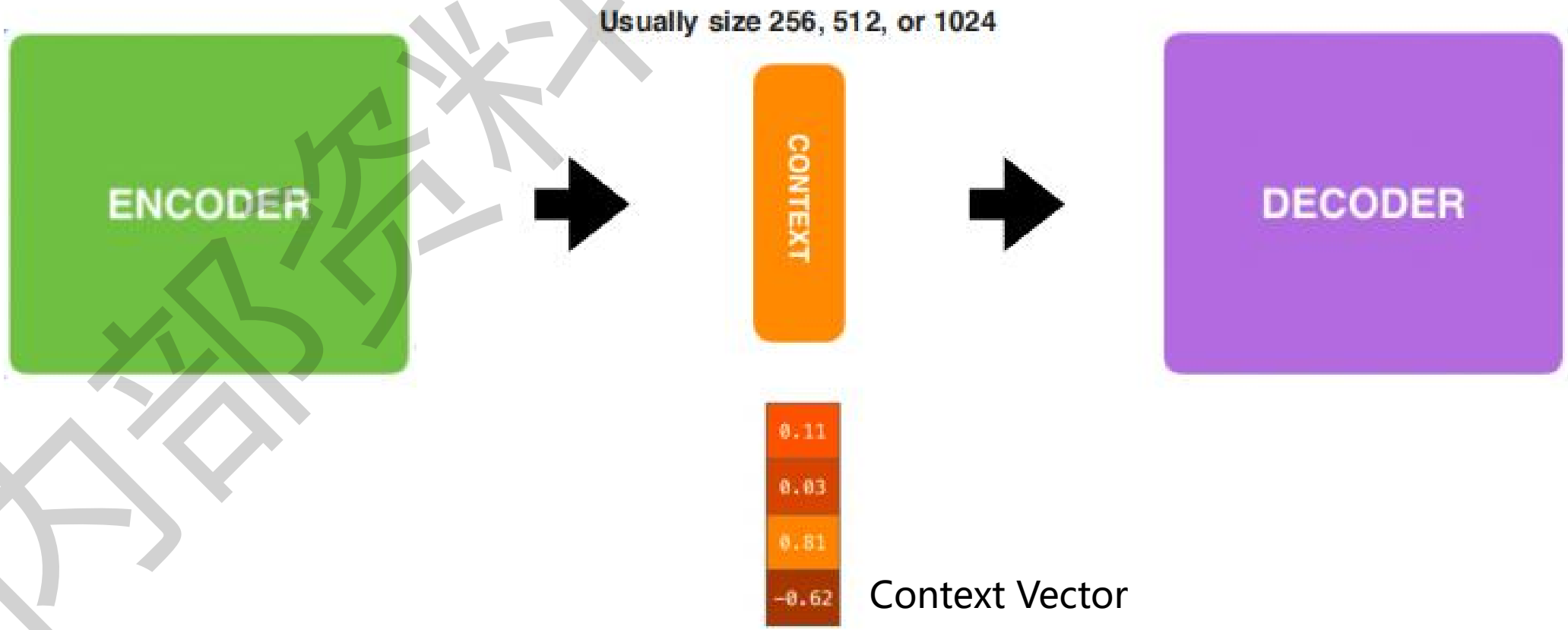
Deep Neural Networks (DNNs) are powerful models that have achieved excellent performance on difficult learning tasks. Although DNNs work well whenever large labeled training sets are available, they cannot be used to map sequences to sequences. In this paper, we present a general end-to-end approach to sequence learning that makes minimal assumptions on the sequence structure. Our method uses a multilayered Long Short-Term Memory (LSTM) to map the input sequence to a vector of a fixed dimensionality, and then another deep LSTM to decode the target sequence from the vector. Our main result is that on an English to French translation task from the WMT'14 dataset, the translations produced by the LSTM achieve a BLEU score of 34.8 on the entire test set, where the LSTM's BLEU score was penalized on out-of-vocabulary words. Additionally, the LSTM did not have difficulty on long sentences. For comparison, a phrase-based SMT system achieves a BLEU score of 33.3 on the same dataset. When we used the LSTM to rerank the 1000 hypotheses produced by the aforementioned SMT system, its BLEU score increases to 36.5, which is close to the previous best result on this task. The LSTM also learned sensible phrase and sentence representations that are sensitive to word order and are relatively invariant to the active and the passive voice. Finally, we found that reversing the order of the words in all source sentences (but not target sentences) improved the LSTM's performance markedly, because doing so introduced many short term dependencies between the source and the target sentence which made the optimization problem easier.





1. 使用预训练向量
2. 使用我们自己的数据进行训练

N vs M (Seq2Seq模型)



THANK YOU

CLOUD.BAIDU.COM