

# LSTM

## RNN

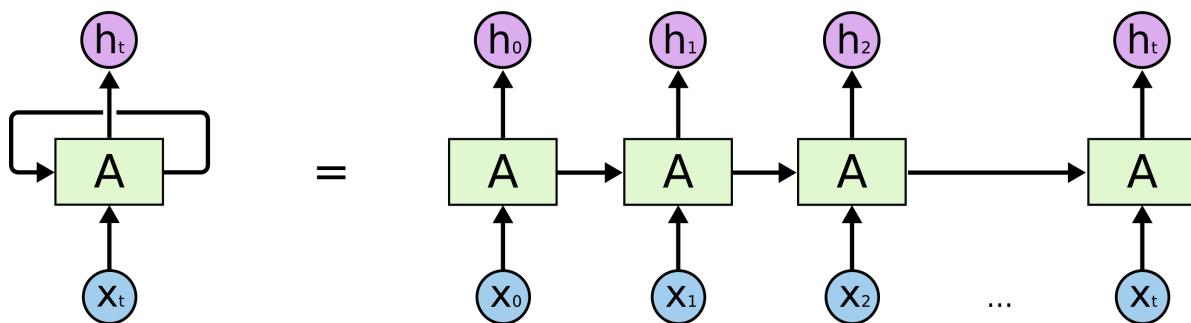
有记忆力的

阅读时会根据先前单词来理解每个单词

但传统的神经网并不能做到这一点

循环神经网络可以看作是同一网络的多个副本，每个副本都将一条消息传递给后继者。

展开的递归神经网络



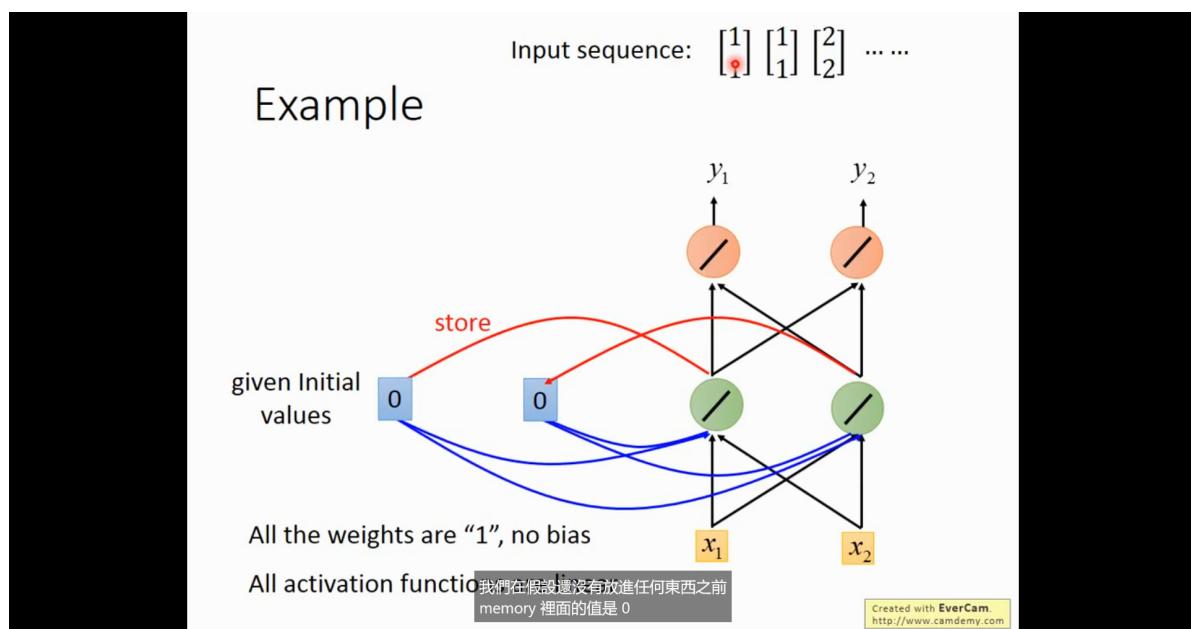
有一个memory用来存储所有隐藏神经元的输出

下次神经元接收输入时,隐藏层会考虑存在memory里的值

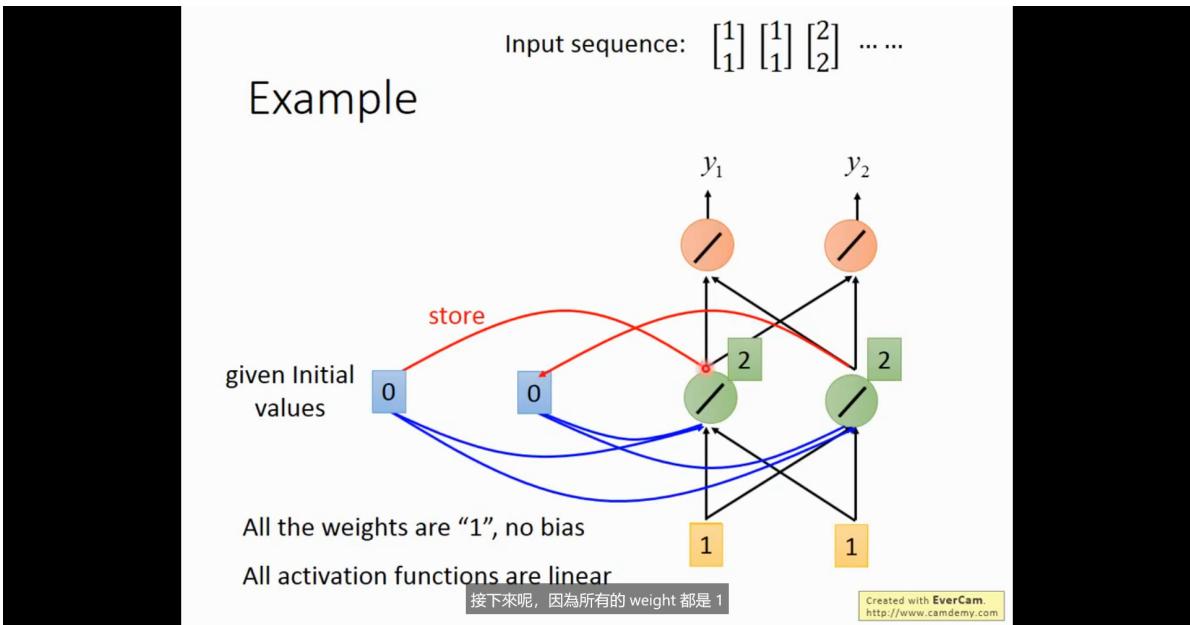
## 例子

假设,边权值都是1,都是liner层,没有bias

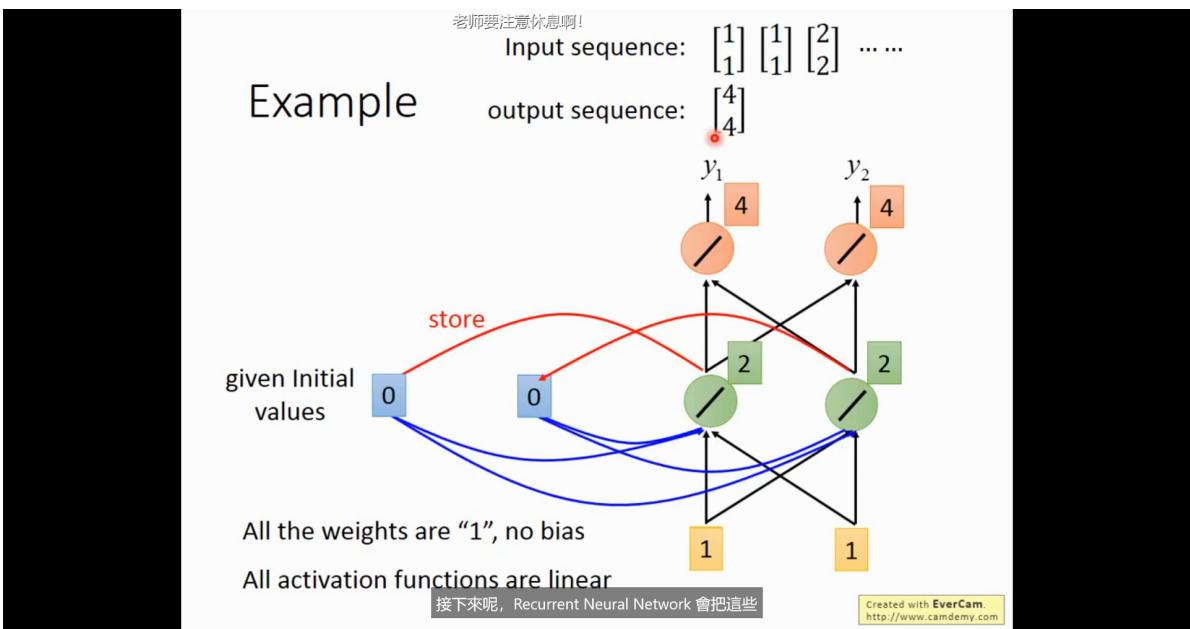
给memory起始值



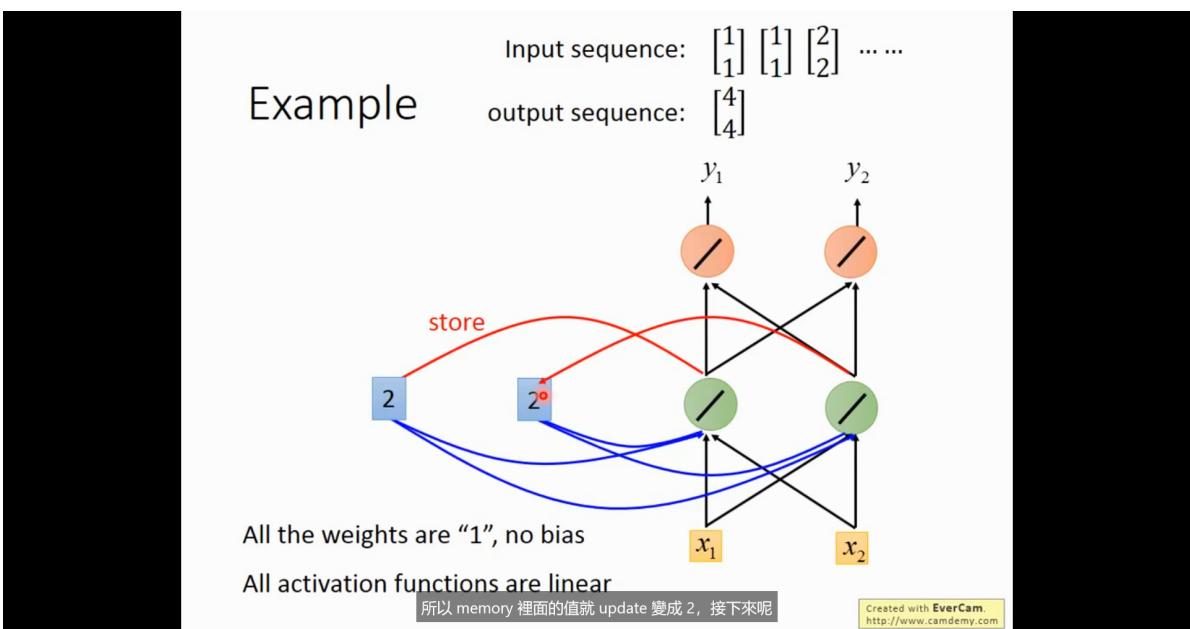
输入值,并向前传递



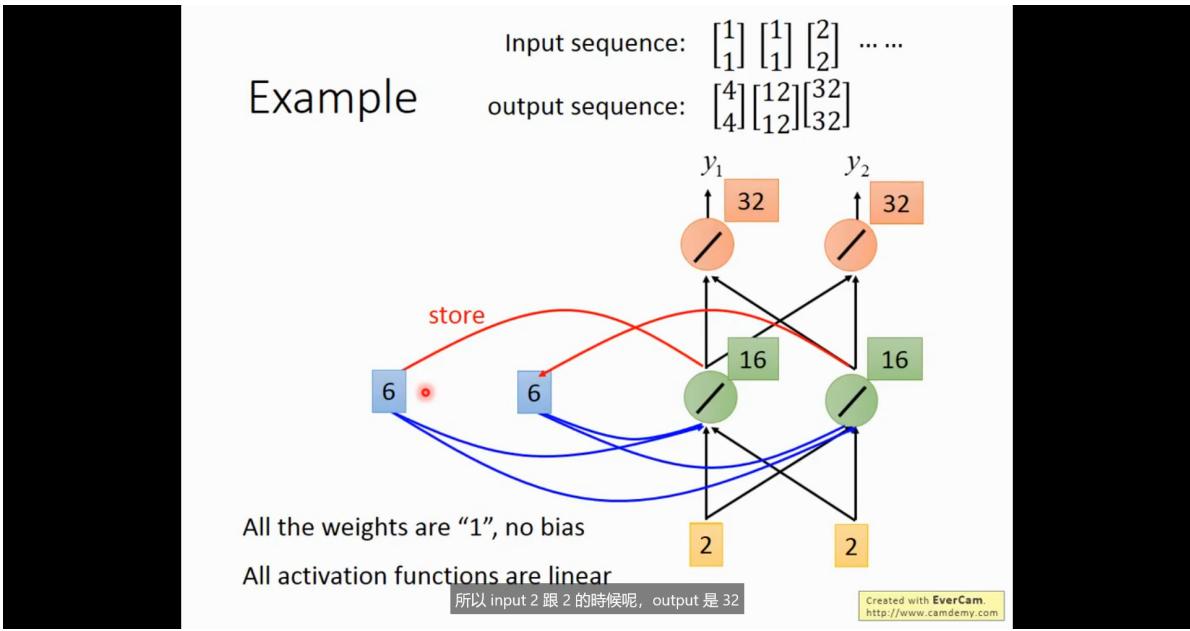
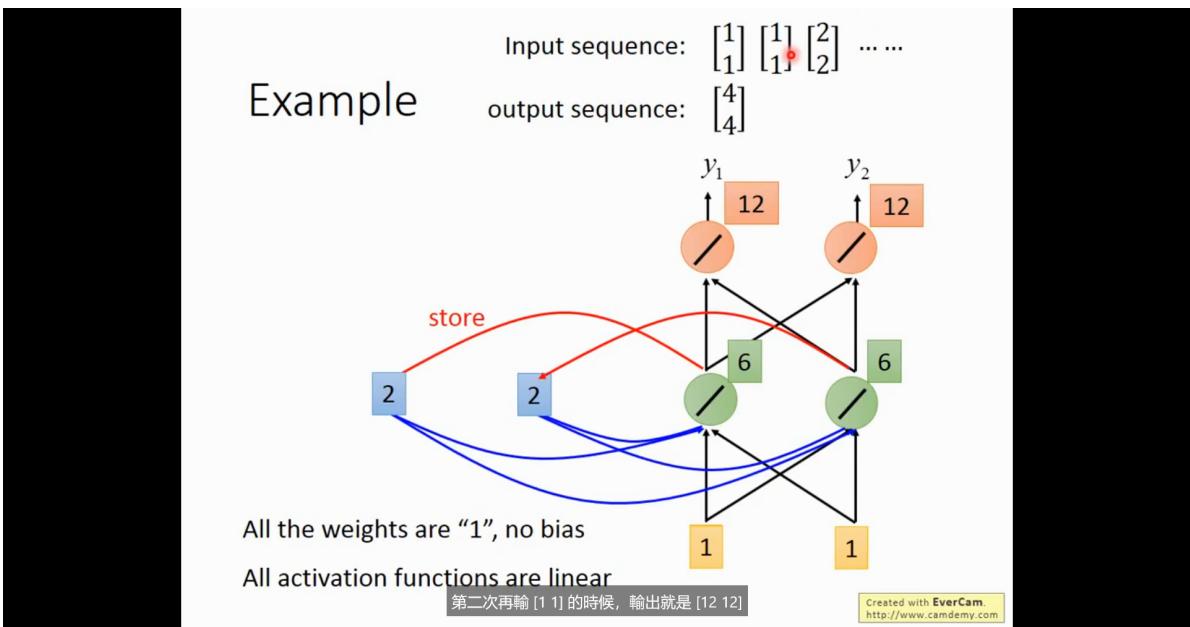
得到output



Memory中的值进行更新

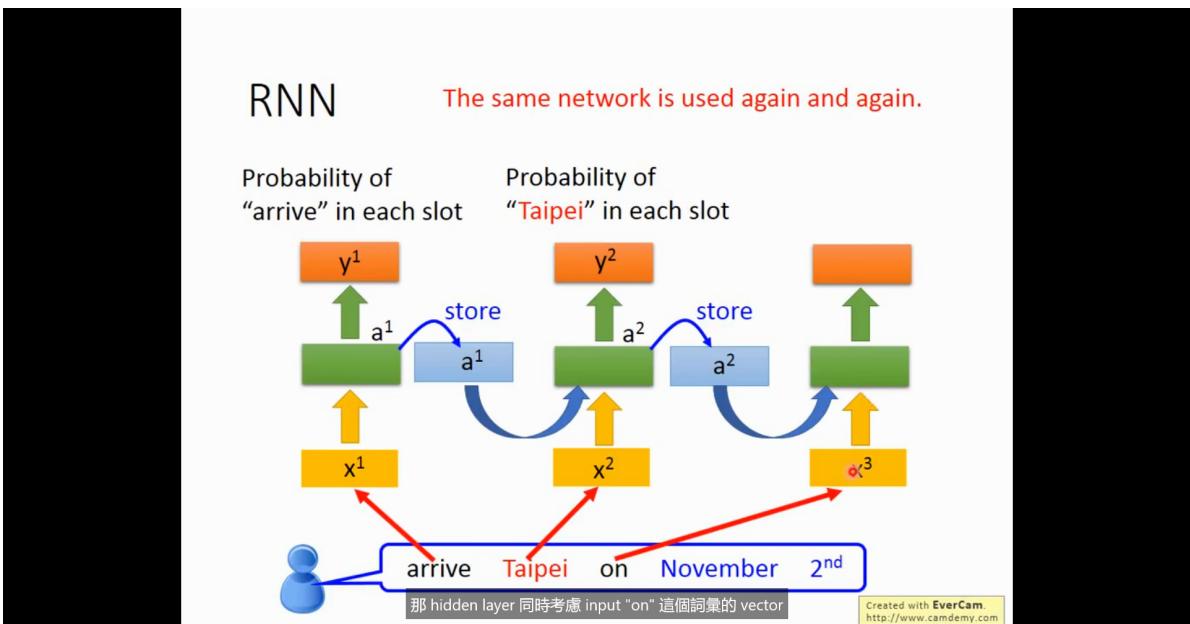


再输入下一个值,并向后更新网络参数,蓝色的线也要算进去,隐藏层的输出6 = 1 + 1 + 2 + 2



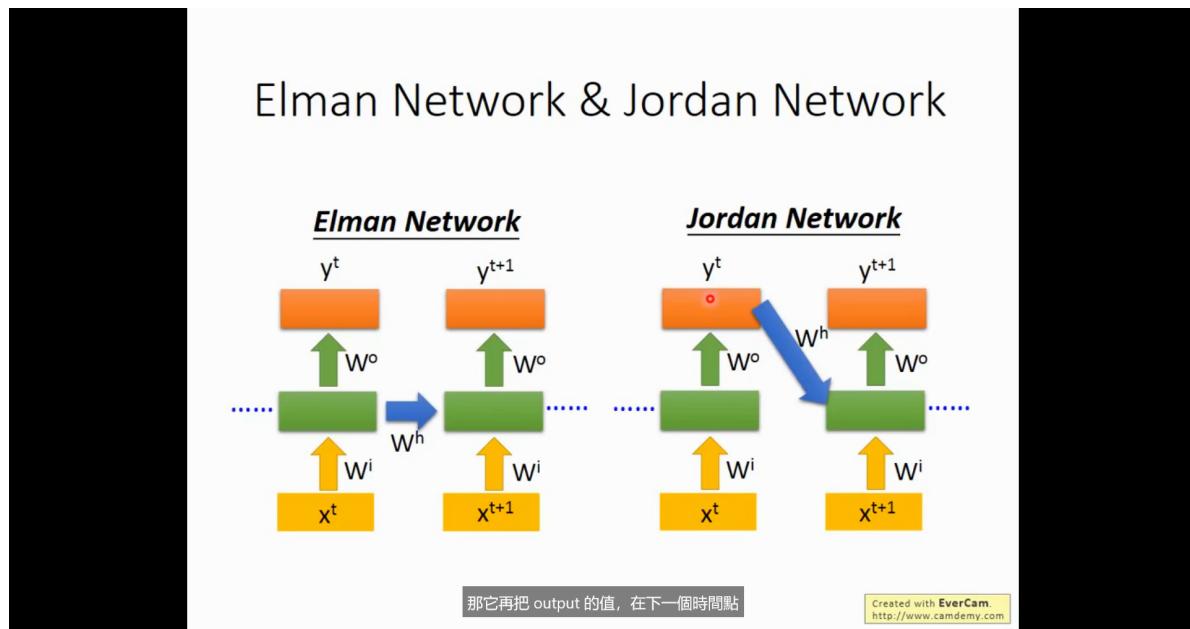
随意的改变输入顺序是会改变输出的值的

## 网络模型



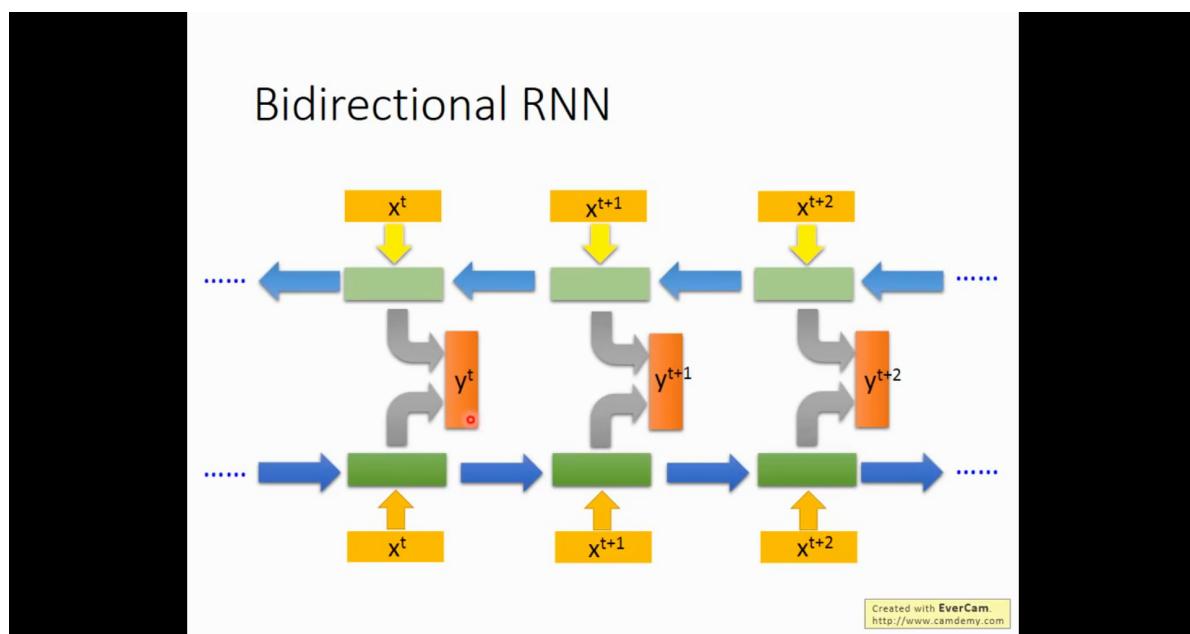
输入一个句子,每次按顺序输入单词,经过传播后输出是这个单词的概率,并将隐藏层参数保存结合下一个单词的输入,计算应该的输出值,以此类推

是同一个网络,在不同的时间(序列输入前后)进行训练



Elman Network把隐藏层的值直接拿过来给下一时间段用

Jordan Network把输出层的值拿来给下一时间段用



也可以正向训练一次,逆向训练一次,再把输出结合

这样看的范围比较广

## LSTM

Long short-term memory

三个门,自己学什么时候打开,什么时候关闭

RNN每次训练完一个时间片,就会更新memory

但是LSTM有三个门去控制,时候会更新memory

## Input Gate

0关 1开

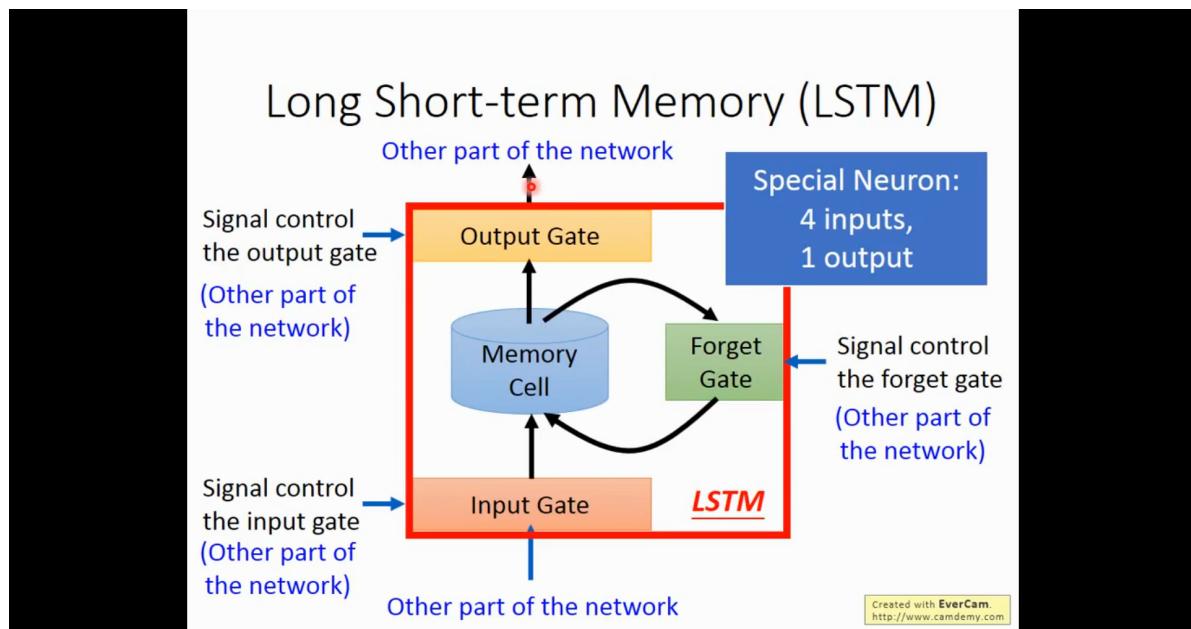
## Output Gate

0关 1开

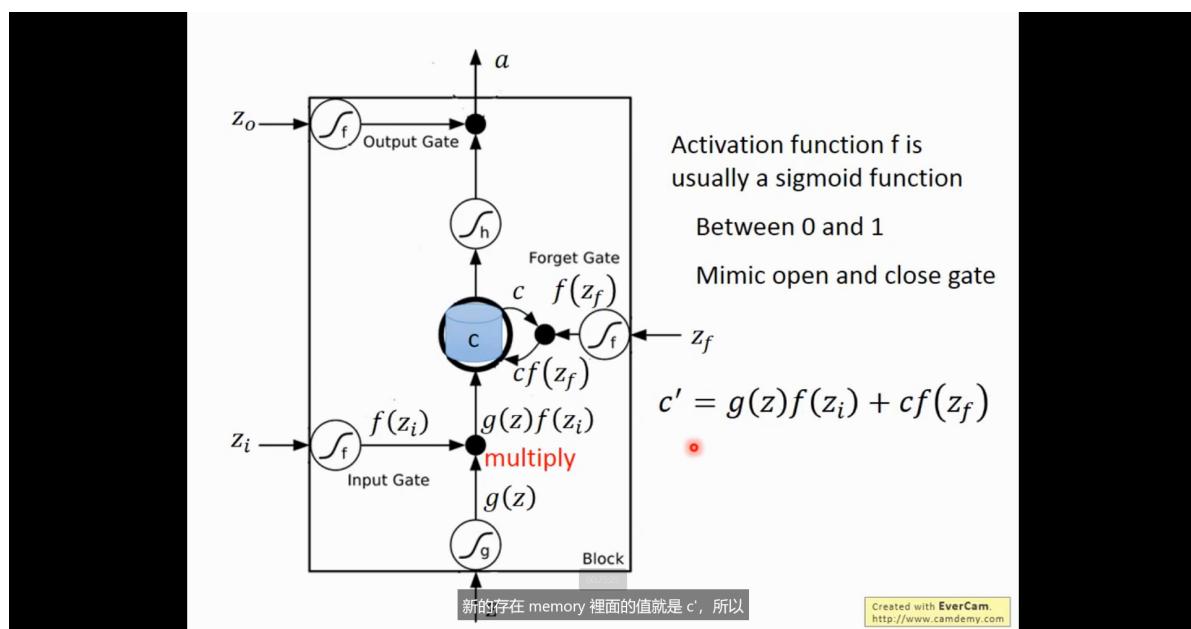
## Forget Gate

0更新 1不更新

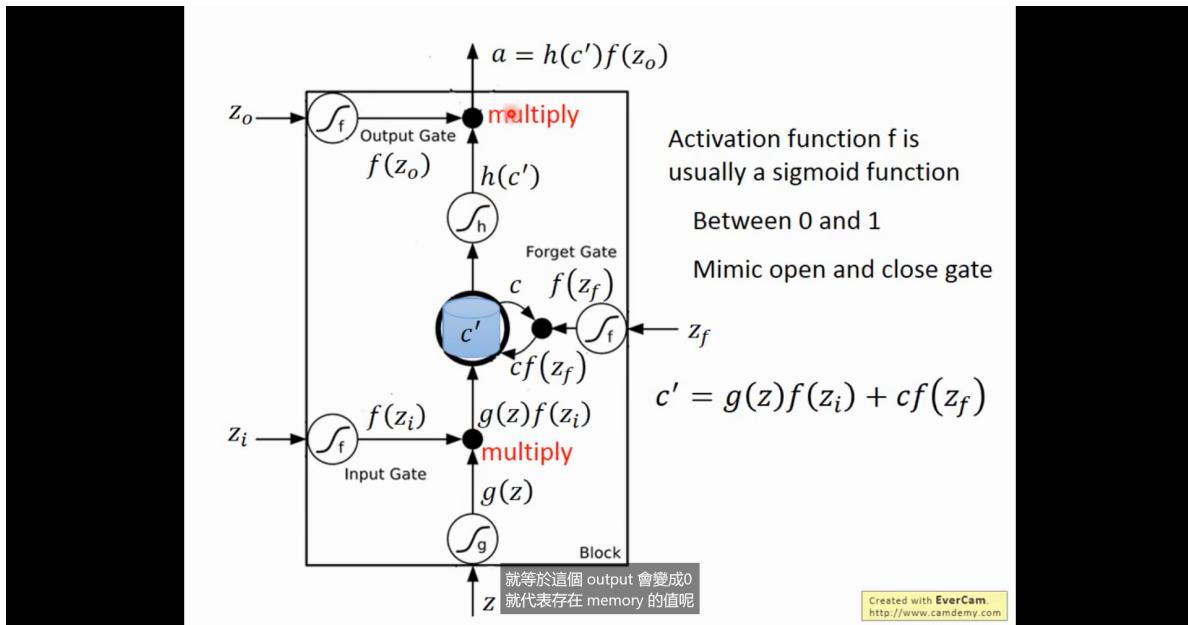
关闭遗忘,打开更新



## 网络模型



一般f多为sigmod函数,因为是0-1之间的数值



例子

## LSTM - Example

$x_1$	0	0	3	3	7	7	7	0	6
$x_2$	1	0	2	1	0	0	-1	0	1
$x_3$	0	0	0	0	0	1	0	0	1
$y$	0	0	0	0	0	7	0	0	6

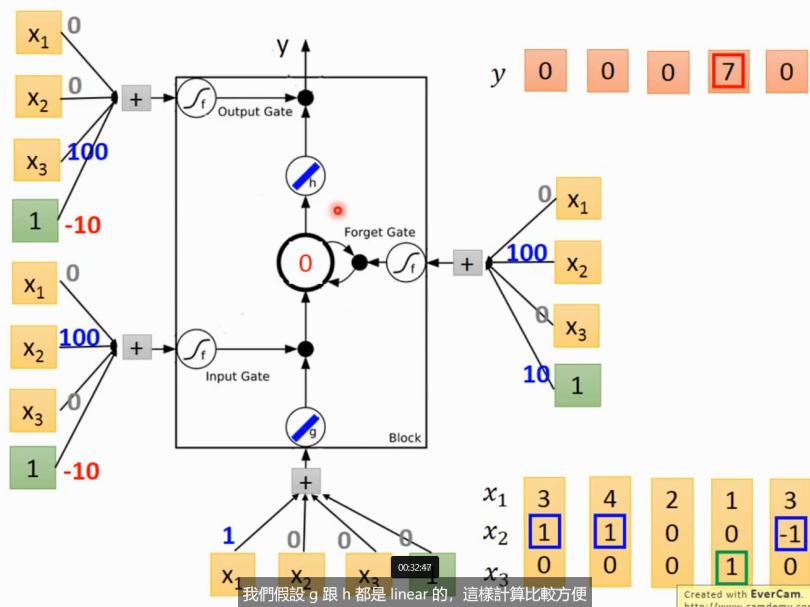
When  $x_2 = 1$ , add the numbers of  $x_1$  into the memory

When  $x_2 = -1$ , reset the memory

When  $x_3 = 1$ , output the number in the memory.

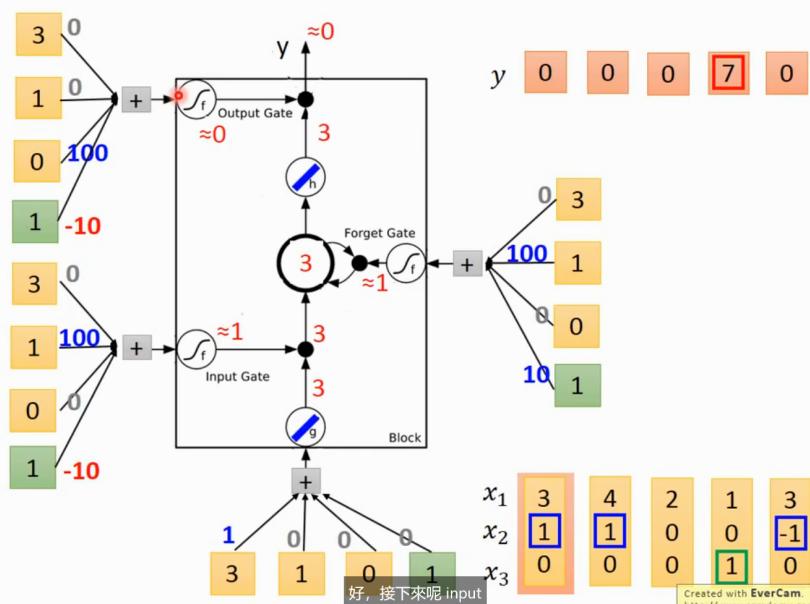
然後看到 1 就會把 6 存進去，所以得到的值是 6

Created with EverCam  
http://www.camdem.com



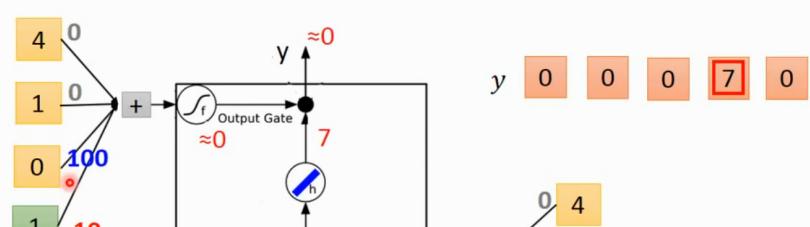
我們假設  $g$  跟  $f$  都是 linear 的，這樣計算比較方便

Created with EverCam  
http://www.camdem.com

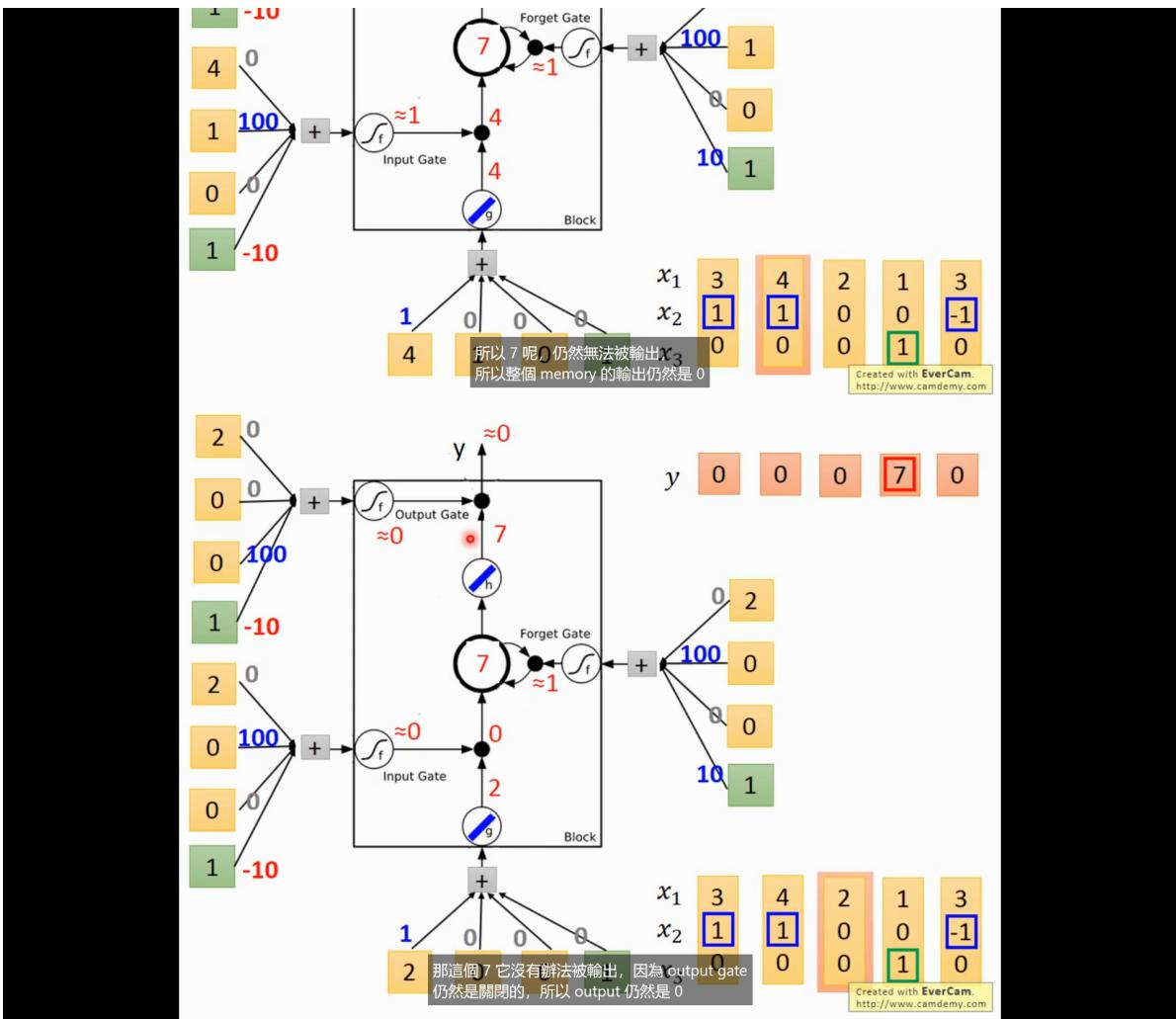


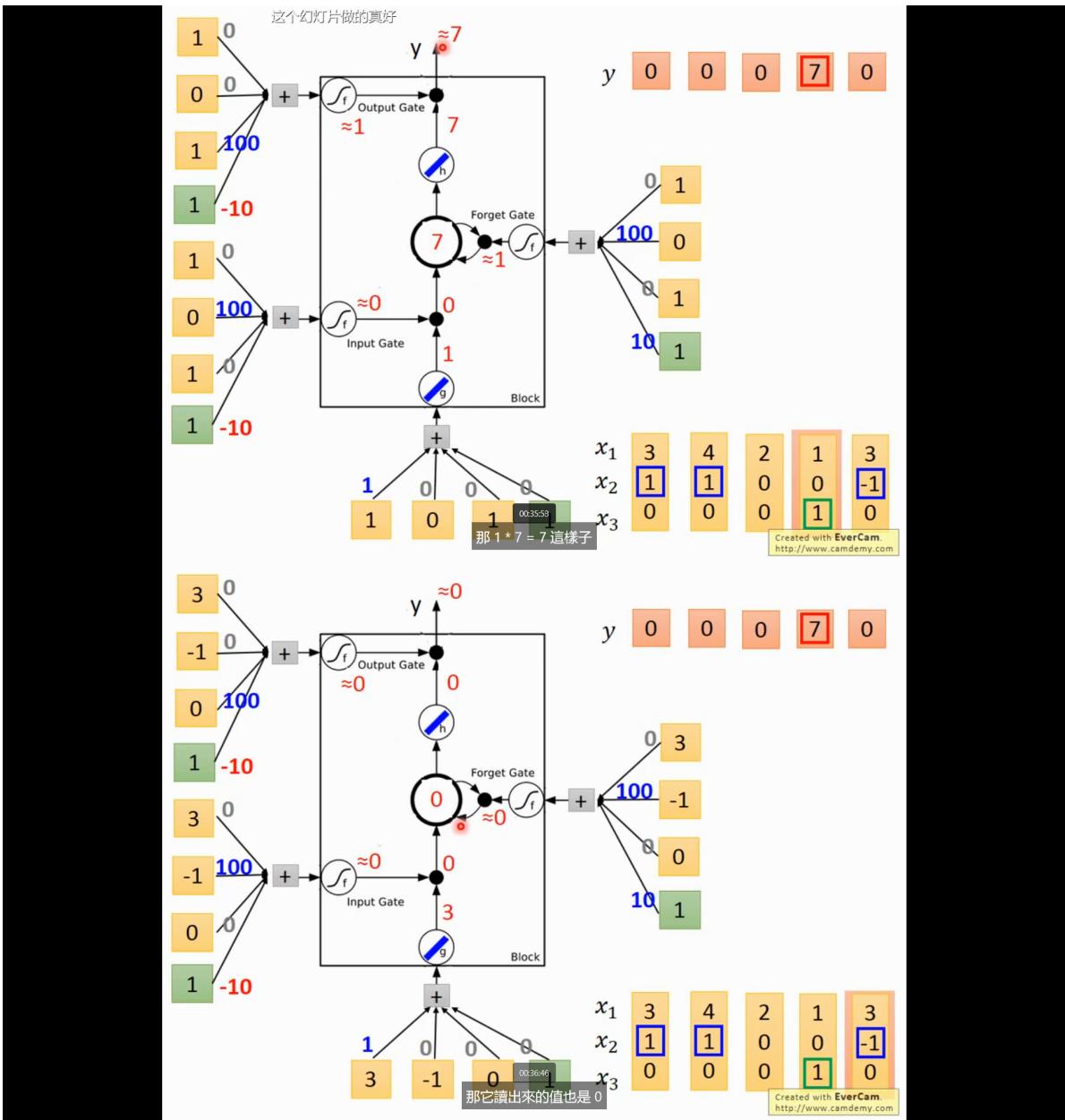
好，接下來呢 input

Created with EverCam  
http://www.camdem.com

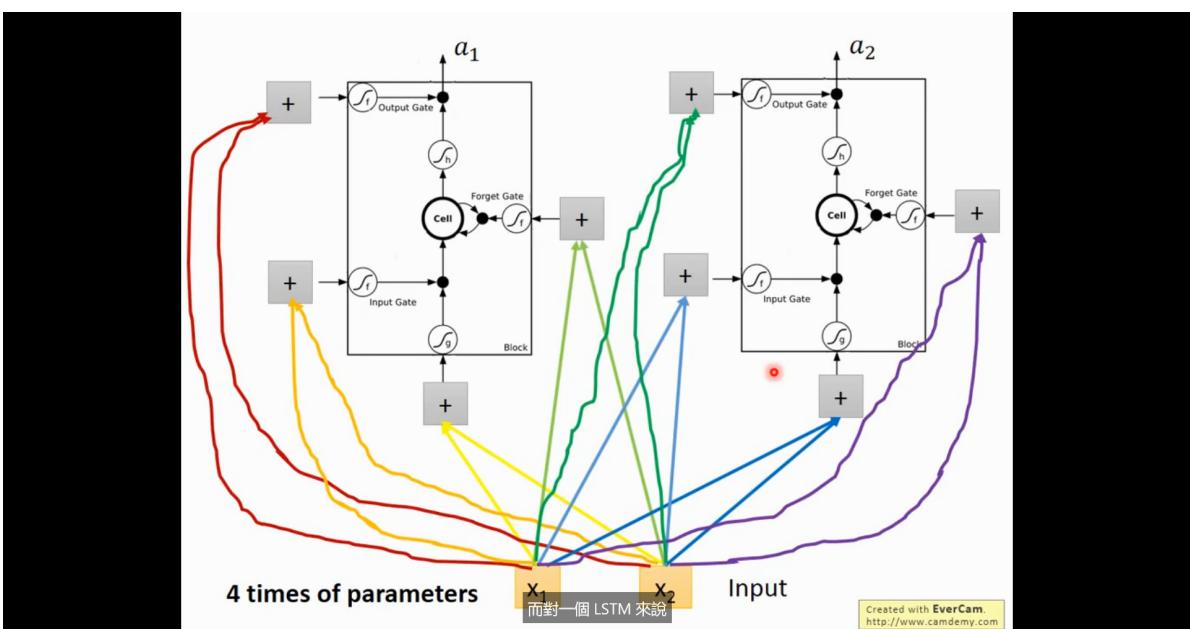


0 4

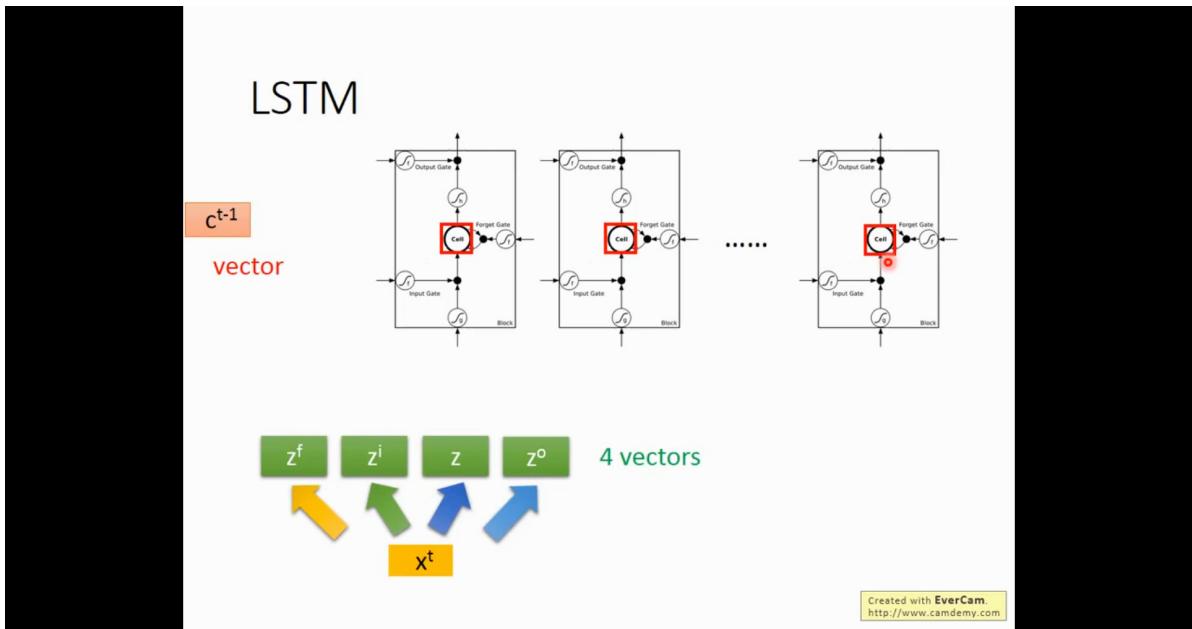




把原来网络中的神经元换成LSTM的memory



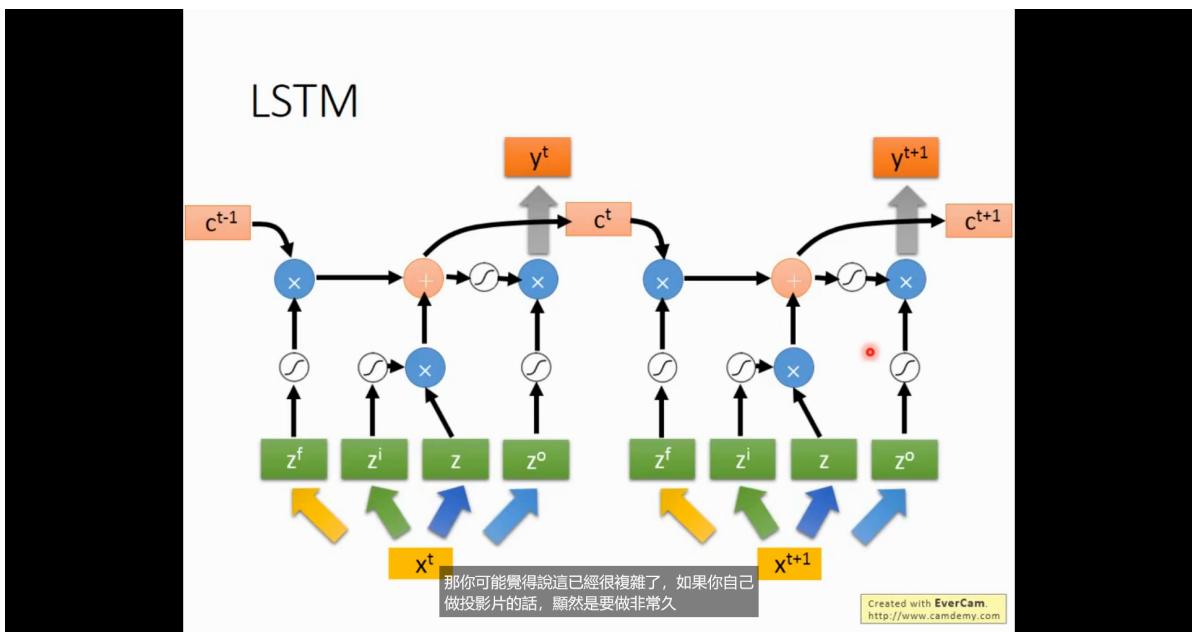
4个input其实是不一样的



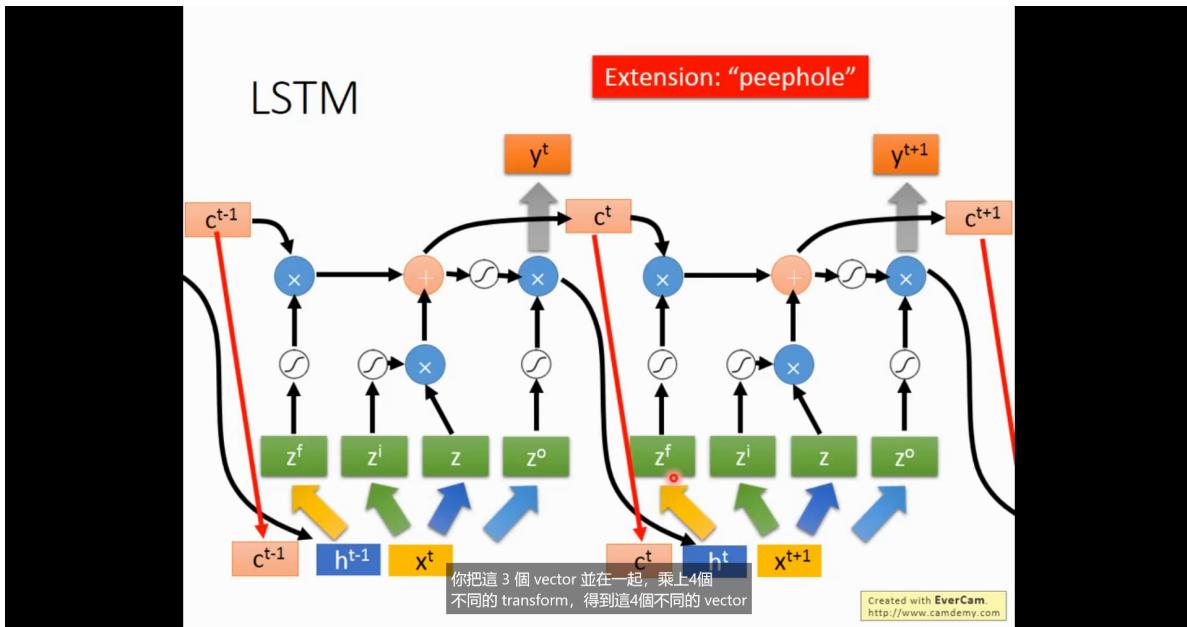
$c^{t-1}$  是每个memory中值合成的一个向量

$z^f, z^i, z, z^o$ 的维度和memory的个数是相同的,分别控制输入和三个门

### 简单LSTM模型

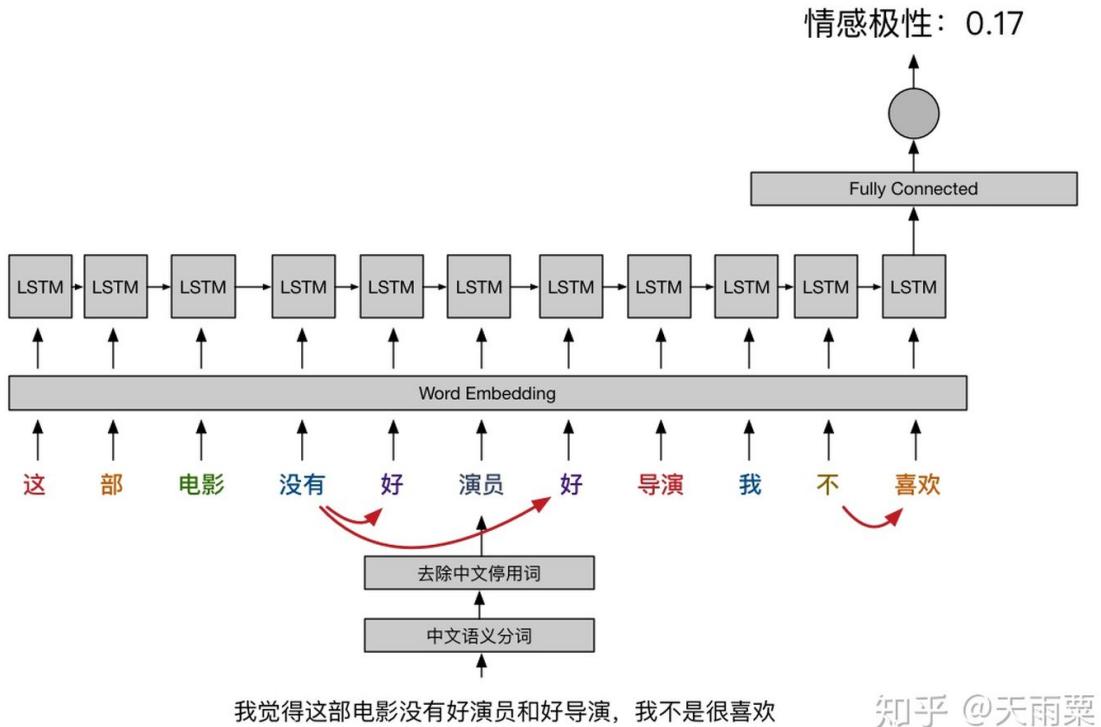


### 真正的LSTM



把 $t-1$ 时刻的memory中的值,和 $t-1$ 时刻的输出合并到 $t$ 时刻的输入,变成一个输入

## NLP上的应用



由于LSTM存在cell state的传递, 如图中LSTM中链接的箭头所示, 它能够捕捉到这种否定关系, 从而输出正确的情感系数。

forget gate里面大多数位置都是0, 少数为1, 这部分1就是它要在网络中持续保留的信息

这里gate有点类似于attention, 对于我所需要的信息, 我就给予高attention (对应为1), 对于无用信息, 我就选择不attention (对应为0)

同理, 如果在某个时刻下信息较为重要, 那么它对应的forget gate位置会一直保留在接近于1的数值, 这样就可以让这个时刻的信息一直往下传递下去而不至于被丢失, 这也是为什么LSTM能够处理长序列的原因之一。

## GRU

GRU比LSTM结构更加简单，只有2个gate，LSTM有3个gate，需要训练的参数更少，具体实现上也更快一点；另外，GRU中只有一个state，它把LSTM中的cell state和activation state视为了一个state。

GRU的两个gate，一个是reset gate，一个是update gate。

reset gate是对t-1时刻状态重置，它也是一个经过sigmoid激活的输出

GRU把update gate既当做update使，又当做forget使