5期-核心BI-week6-12月12日24点截止

*1. 与CNN相比，RNN的区别是?

- A.CNN在空间上扩展，RNN在时间上扩展
- B. RNN具有记忆功能，CNN没有记忆功能
- C.RNN可以输出时间上的连续状态，而CNN是静态输出
- D.以上都是

*2. 下列方法中不能解决梯度消失的是?

- A.避免使用sigmoid函数，选择更好的激活函数，如Relu激活函数
- B.加入BN层
- C.权重正则化
- D.以上都不能

*3. 下列有关LSTM说法错误的是?

- A.LSTM包含记忆门，遗忘门和输出门
- B.相较于RNN，LSTM可以解决梯度消失问题
- C.随着序列变长，LSTM效果会越来越好
- D.以上全错误

*4. 下列有关GRU说法错误的是?

- A.相较于LSTM，GRU 参数更少
- B.相较于LSTM，是数据集很大的情况下，GRU表达能力更好
- C.GRU包含更新门，重置门
- D.以上全错误

*5. 下列关于Batch Normalization 的说法正确的?

- A.对数据做批规范化
- B.是的模型变得更复杂
- C.对模型权重归一化
- D.以上均不是

*提问1:

问题需为与本节课程内容相关的技术问题。暂无或与课程内容无关均判定为不符合要求

对于随机梯度下降来说，由于是随机丢弃，故而每一个mini-batch都在训练不同的网络。每一个mini-batch都在训练不同的网络是怎么理解？？？

*提问2:

问题需为与本节课程内容相关的技术问题。暂无或与课程内容无关均判定为不符合要求

CNN与RNN区别？？？

*提问3:

问题需为与本节课程内容相关的技术问题。暂无或与课程内容无关均判定为不符合要求

GRU的应用场景?

使用**LSTM** 进行**手写字体识别**

作业

数据切割出一份**训练集**，一份**验证集**。

LSTM第一层接**32个神经元**

第一层lstm 后接一个**dropout0.2**

LSTM第二层接**32个**神经元

第二层lstm 后接一个**dropout0.3**

需使用**callbacks函数**分别用到**earlystop**，**ModelCheckpoint**，**ReduceLROnPlateau**

使用**load_weight**的形式导入以上训练的模型，并对**验证集**进行**预测**

CNN和RNN区别

CNN主要用于图像；RNN主要用于时序和NLP。

当CNN、RNN都用于NLP时，它们的区别在于：

RNN（循环神经网络），当前节点的输入包含之前所有节点信息。

RNN（递归神经网络），当前节点的输入以树结构形式包含部分之前节点信息。

CNN（卷积神经网络），当前节点的输入以树结构形式仅包含上一层节点信息。

epoch

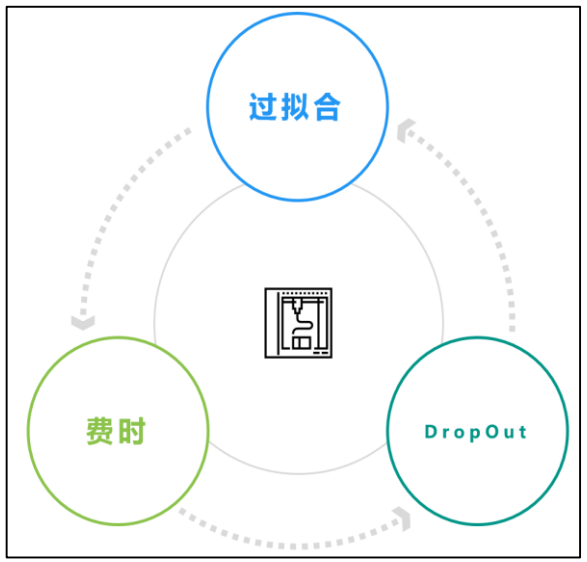
1. 一个epoch指代所有的数据送入网络中**完成一次前向计算及反向传播的过程**

2. Batch Size

Batch就是**每次送入**网络中训练的**一部分数据**，而Batch Size就是**每个batch**中**训练样本的数量**

3. Iterations

iterations就是**完成一次epoch**所需的**batch个数**



keras模型的保存
DROPOUT

- Keras模型的保存
- dropout的用法
- callbacks
- cnn&pooling
- rnn
- lstm

Lession6学习目标

Keras进一步学习
- keras模型的保存
- dropout的用法
- callbacks的用法
- cnn-pooling的用法

RNN&LSTM
- RNN入门
- RNN示意图
- LSTM入门
- LSTM正向传播示意

在深度学习当中，如果训练集过小，神经元过多，或者层数过大，容易产生过拟合现象

具体表现：模型在训练集中损失函数较小，但是在测试集损失函数较大

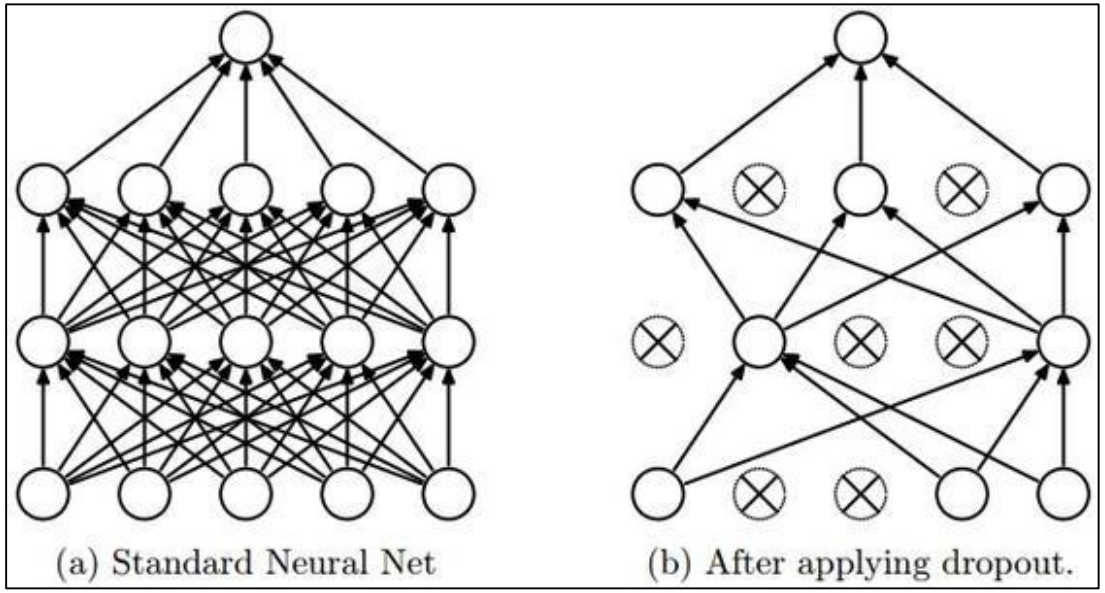
dropout是指在深度学习网络的训练过程中，对于神经网络单元，按照一定的概率将其暂时从网络中丢弃。注意是暂时，对于随机梯度下降来说，由于是随机丢弃，故而每一个mini-batch都在训练不同的网络。

Dropout的步骤

1.**随机删除**（**临时**）网络中**一定比例**的**隐藏神经元**，**输入输出保持不变**；

2.让**输入通过修改后的网络**。然后把**得到的损失**同时**修改后的网络**进行**反向传播**。**在未删除的神经元上面进行参数更新**

3.重复该过程（恢复之前删除掉的神经元，以一定概率删除其他神经元。**前向传播**、**反向传播更新参数**）



keras -Sequential模型-dropout

```
#add 模式
model = Sequential()
model.add(Dense(64, input_shape=(784,)))
model.add(Dropout(0.5))
model.add(Activation('relu'))
model.add(Dense(10))
model.add(Activation('softmax'))
```

Callbacks

回调函数是一组在训练的**特定阶段**被调用的函数集，你可以使用回调函数来**观**察训练过程中**网络内部的状态和统计信息**。

通过**传递回调函数列表到模型.fit()**中，即可在给定的训练阶段调用该函数集中的函数。

callbacks-earlystop

1.**monitor**: 监控的数据接口，有'acc','val_acc','loss','val_loss'等等。正常情况下如果有验证集，就用'val_acc'或者'val_loss'。

2.**min_delta**：增大或减小的阈值，只有大于这个部分才算作improvement。

3.**patience**：能够容忍多少个epoch内都没有improvement。

4.**mode**: 就'auto', 'min',',max'三个可能。如果知道是要上升还是下降，建议设置一下。

```
import keras

early_stopping=keras.callbacks.EarlyStopping(monitor='val_loss', min_delta=0,
                                             patience=0, verbose=0, mode='auto',
                                             baseline=None, restore_best_weights=False)

model.fit(callbacks = [early_stopping])
```

callbacks-ModelCheckpoint

1.filename：字符串，保存模型的路径

2.monitor：需要监视的值

3.verbose：信息展示模式，0或1

4.save_best_only：当设置为True时，将只保存在验证集上性能最好的模型

5.mode:'auto','min','max'之一，在save_best_only=True时决定性能最佳模型的评判准则，例如，当监测值为val_acc时，模式应为max，当检测值为val_loss时，模式应为min。在auto模式下，评价准则由被监测值的名字自动推断。

7.save_weights_only：若设置为True，则只保存模型权重，否则将保存整个模型（包括模型结构，配置信息等）

```
import keras
callbacks = [EarlyStopping(monitor='val_loss', patience=8),
             ModelCheckpoint(filepath='./best_model.h5', monitor='val_loss',
save_best_only=True)]
history=model.fit(X_train, y_train,epochs=40,callbacks=callbacks,
batch_size=32,validation_data=(X_test,y_test))
```

callbacks-ReduceLROnPlateau

1.monitor：被监测的量

2.factor：每次减少学习率的因子，学习率将以**lr = lr*factor**的形式被减少

3.patience：当patience个epoch过去而模型性能不提升时，学习率减少的动作会被触发

3.mode:'auto','min'，'max'之一，在min模式下，如果检测值触发学习率减少。在max模式下，当检测值不再上升则触发学习率减少。

4.epsilon：阈值，用来确定**是否进入检测值的“平原区”**

5.cooldown：学习率减少后，会经过cooldown个epoch才重新进行正常操作

min_lr：学习率的下限

```
import keras

keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.1,
patience=10, verbose=0, mode='auto', epsilon=0.0001, cooldown=0,
min_lr=0)
```

callbacks-CSVLogger

1.iename：保存的csv文件名，如run/log.csv

2.separator：字符串，csv分隔符

3.append：默认为False，为True时csv文件如果存在则继续写入，为False时总是覆盖csv文件

CNN&Pooling

卷积神经网络-背景

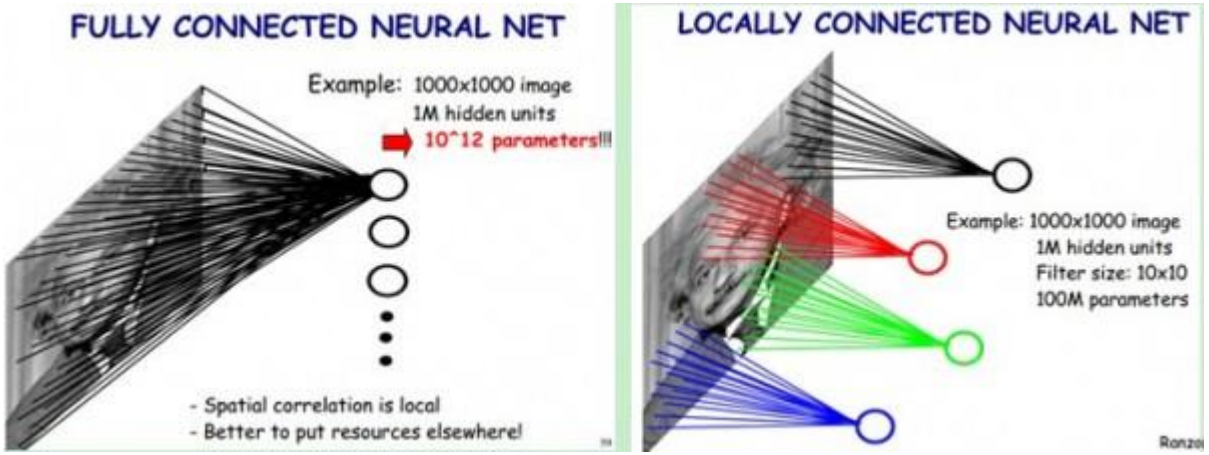
往往把图像表示为像素的向量，比如一个**1000×1000**的图像，可以表示为一个**1000000**的**向量**。**全连接神经网络**中，如果**隐含层数目**与**输入层一样**，即也是1000000时，那么输入层到隐含层的**参数**数据为**1000000×1000000=10^12**

卷积神经网络-局部感知

人对外界的认知是从**局部**到**全局**的，而图像的空间联系也是**局部的像素联系**较为紧密，而**距离较远的像素相关性则较弱**。

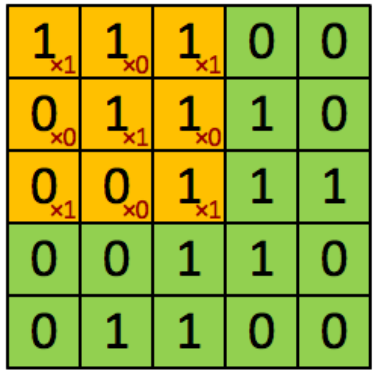
因而，每个神经元其实没有必要对全局图像进行感知，只需要对局部进行感知，然后在**更高层**将**局部**的**信息综合**起来就**得到了全局的信息**。

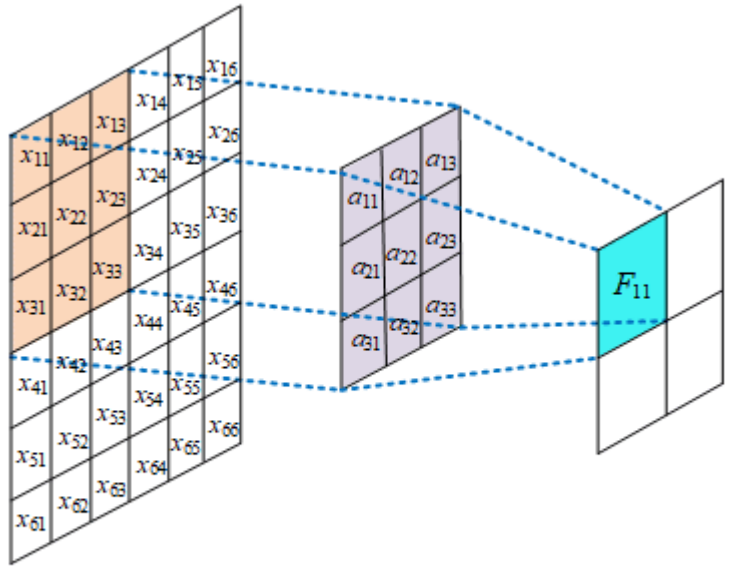
假如每个神经元只和10×10个像素值相连，那么权值数据为1000000×100个参数，减少为原来的万分之一。而那10×10个像素值对应的10×10个参数，其实就相当于卷积操作。



卷积神经网络-权值共享

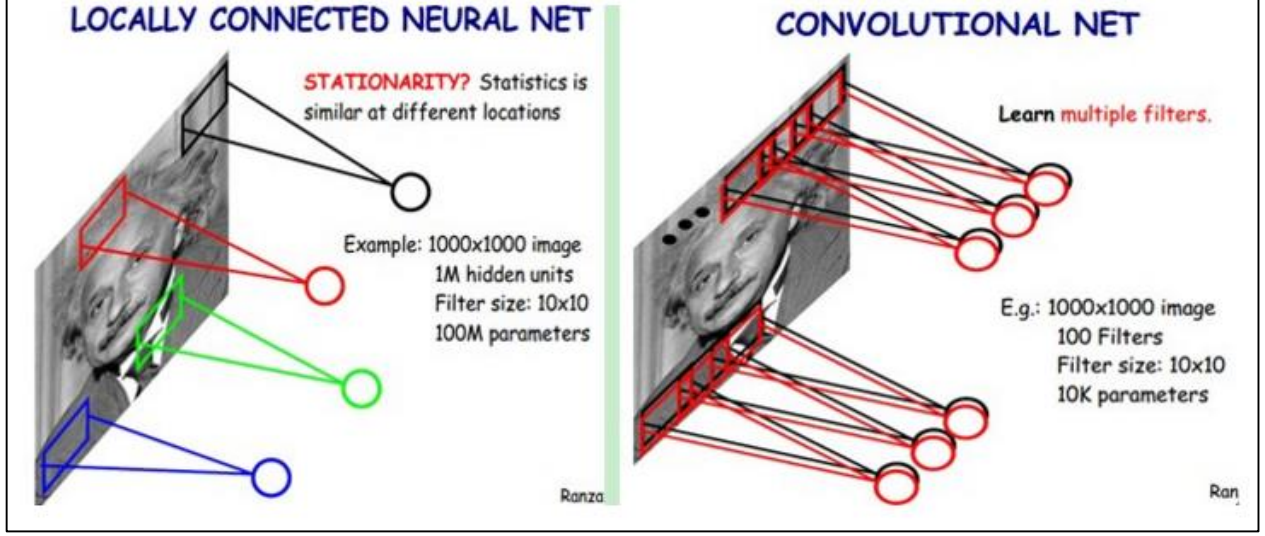
局部连接中，每个神经元都对应10X10个参数，一共1000000个神经元，如果这1000000个神经元的100个参数都是相等的，那么参数数目就变为100了。





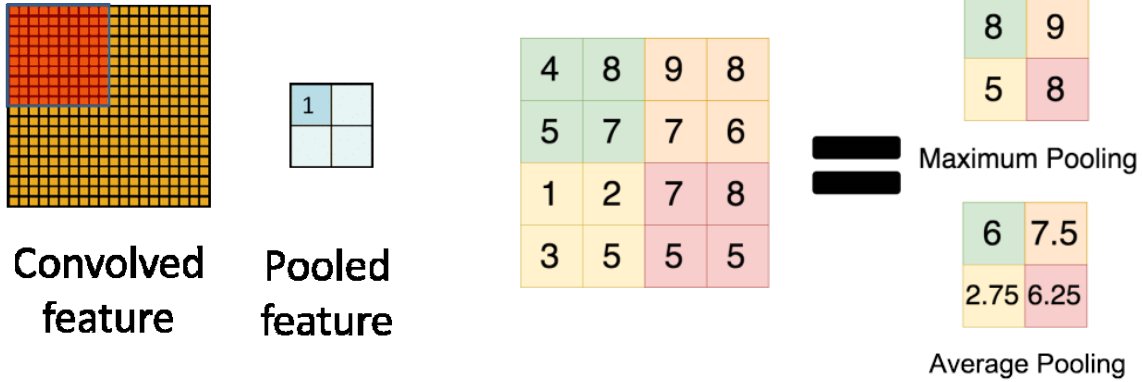
卷积神经网络-多个卷积核

一种**滤波器**，也就是**一种卷积核**就是提出**图像的一种特征**，例如某个方向的边缘。那么我们需要提取不同的特征，怎么办，加多几种滤波器不就行了吗？对了。所以假设我们加到100种滤波器，每种滤波器的参数不一样，表示它提出输入图像的不同特征，例如不同的边缘。这样**每种滤波器去卷积图像就得到对图像的不同特征的放映**，我们称之为**Feature Map**。所以100种卷积核就有100个Feature Map。这100个Feature Map就组成了**一层神经元**。到



pooling

pooling是在卷积网络（CNN）中一般在**卷积层**（conv）**之后**使用的**特征提取层**，使用pooling技术将卷积层后得到的**小邻域内的特征点整合得到新的特征**。一方面**防止无用参数增加时间复杂度**，一方面**增加了特征的整合度**。



Max &Mean Pooling

Pooling：整个图片被**不重叠**的分割成**若干个同样大小的小块**（pooling size）

Max Pooling:**最大池子化**，池化（max-pooling）即**对局部接受域中的所有值最大值**。

Mean Pooling：**均值池化**（mean-pooling）即对局部接受域中的所有值求均值。

RNN的背景　语言模型

RNN在**自然语言处理**领域用的比较**热**。

example:本书全面介绍了统计自然语言处理的基本概念。

语言模型：给定一个**一句话前面**的部分，**预测接下来**最有可能的一个词**是什么**。

循环神经网络语言模型

好的语言模型应该至少**捕捉自然语言**的**两个特征**：**语法特性**与**语义特性**。为了保证语法的正确性，我们往往只需要考虑**生成词的上下文**；这也就意味着**语法**的特性往往属于**局部特**性。而**语义的一致性**则**复杂**了许多。我们需要考虑大量的乃至于整个**文档语料集**的**上下文**信息来**获取正确的全部语义**

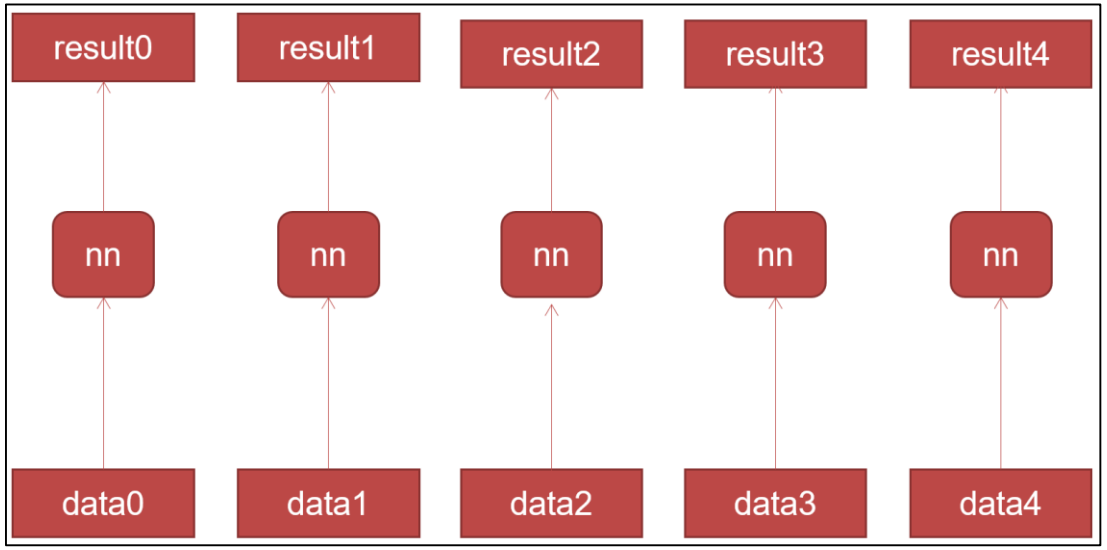
01

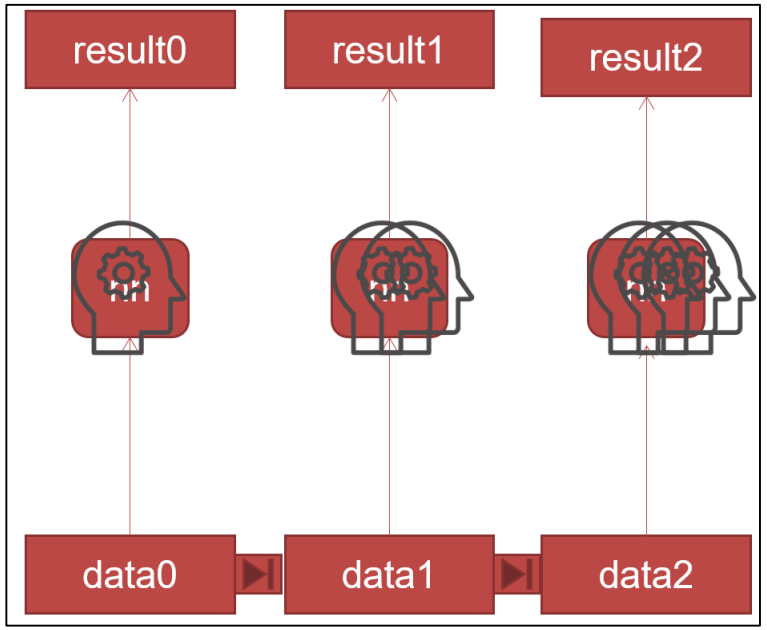
什么是RNN1

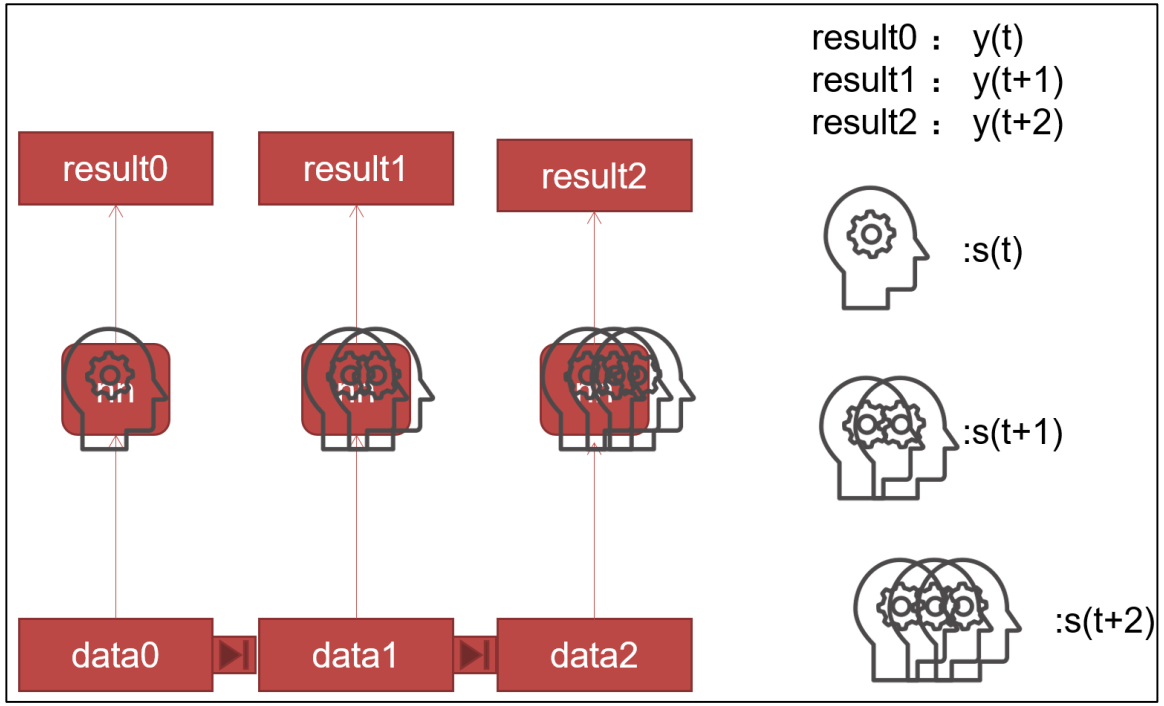
在传统的神经网络中，假设了所有的输入（包括输出）之间是相互独立的。对于很多任务来说，这是一个非常糟糕的假设。

我教你上语文课，其中，提到了有关于数学的问题。

你教我上数学课，其中，提到了有关于语文的问题。







02

什么是RNN2

如何让nn记住data0到data2他们之间是有关联的

03

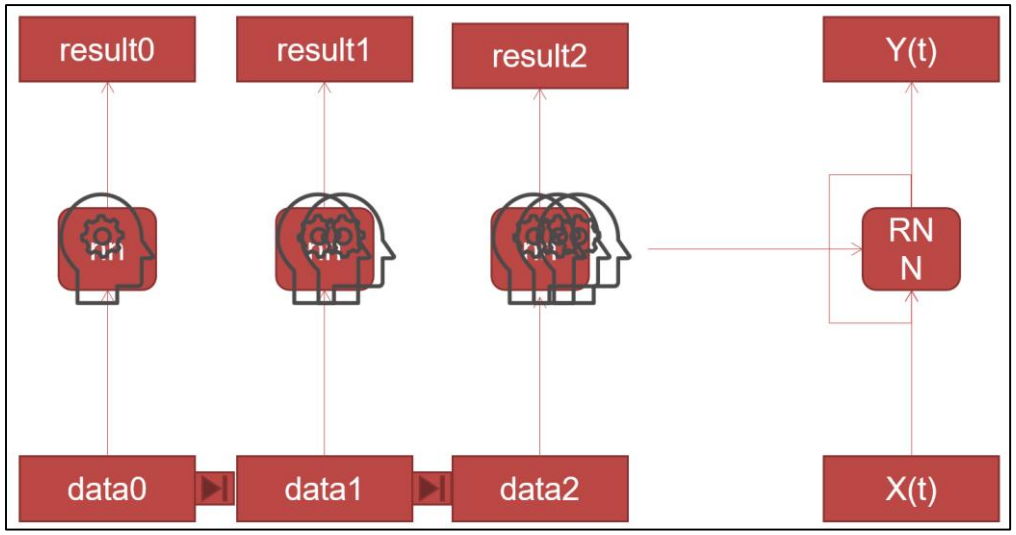
什么是RNN3

如何让nn记住data0到data2他们之间是有关联的

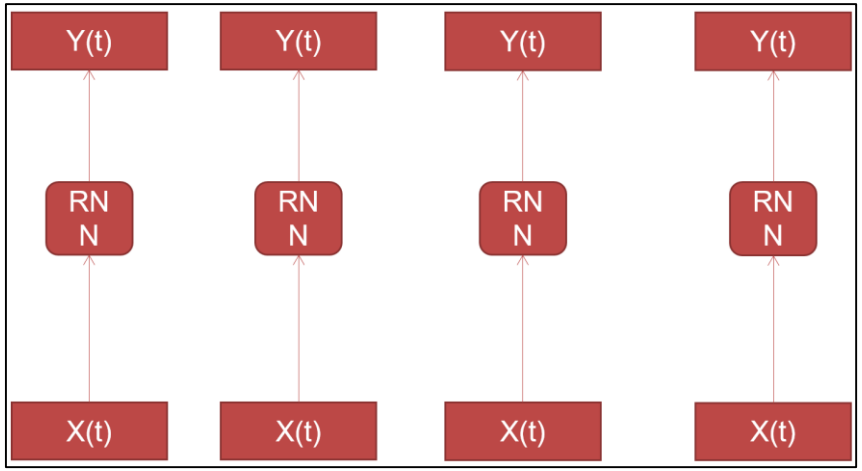
04

什么是RNN4

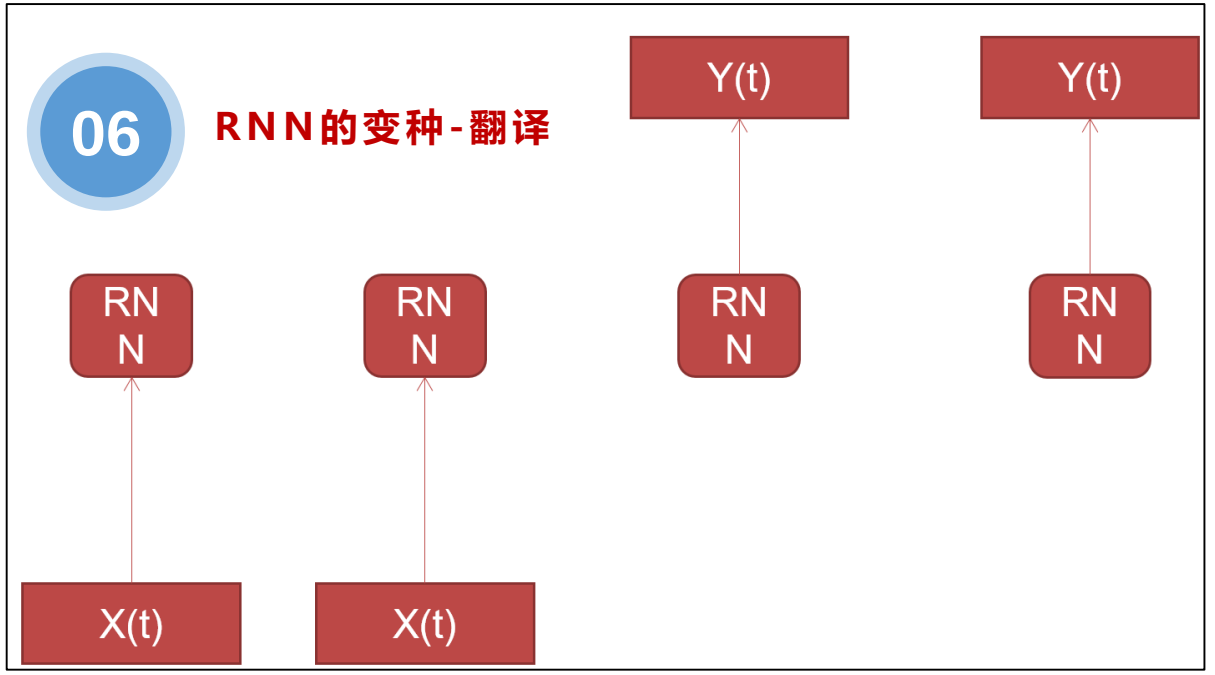
如何让nn记住data0到data2他们之间是有关联的



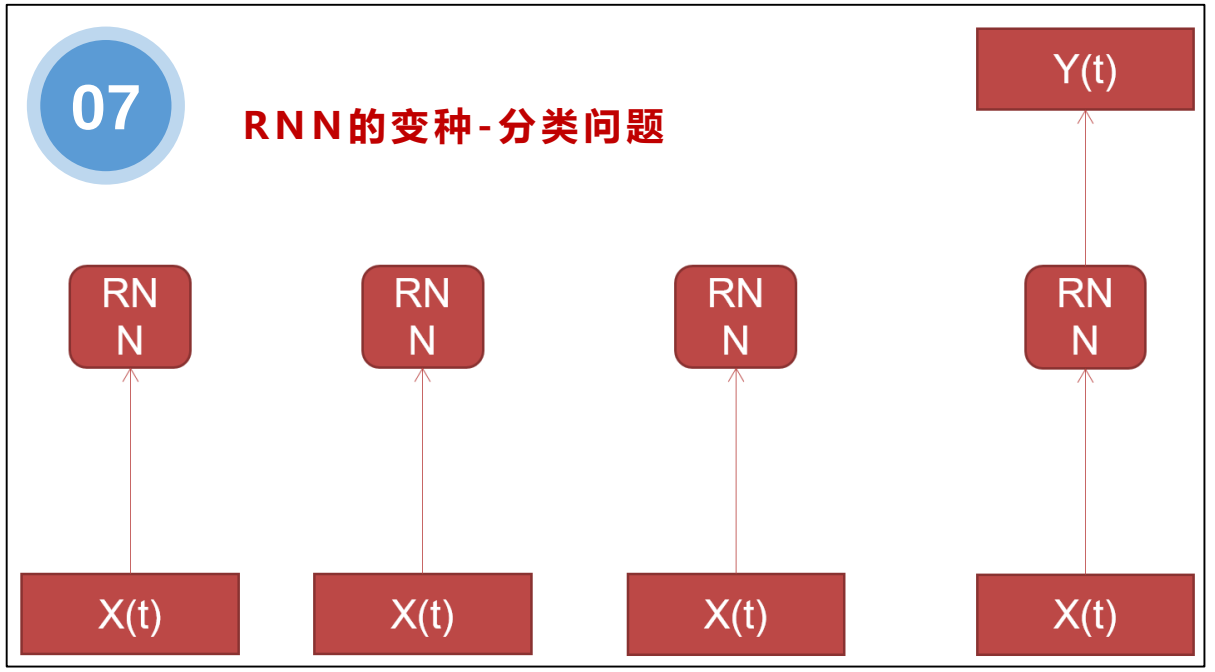
05 RNN的变种

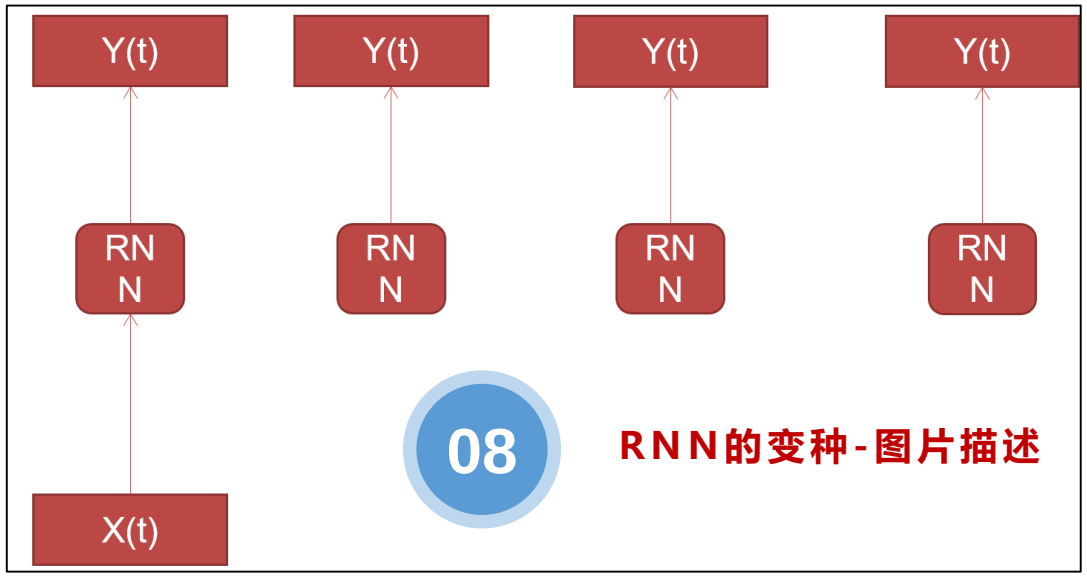


06 RNN的变种-翻译



07 RNN的变种-分类问题





08 RNN的变种-图片描述

RNN正向向传播示意

01 RNN正向传播示意

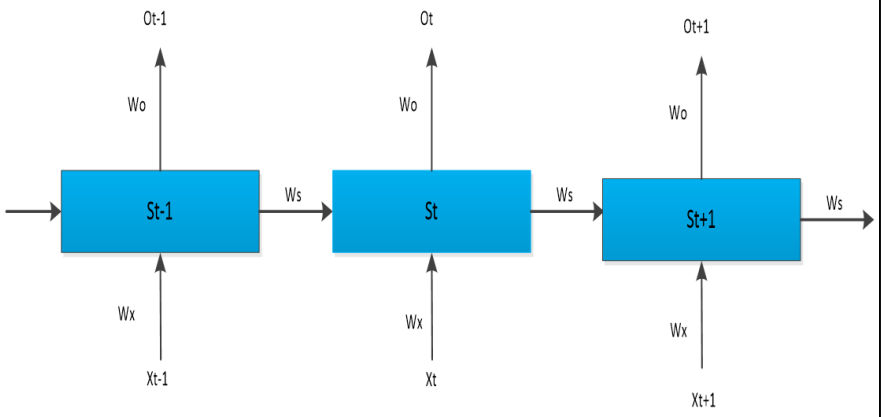
$$S_1 = w_x x_1 + w_s S_0 + b_1 \quad S_1 = \sigma(S_1) \quad O_1 = W_O S_1 + b_2$$

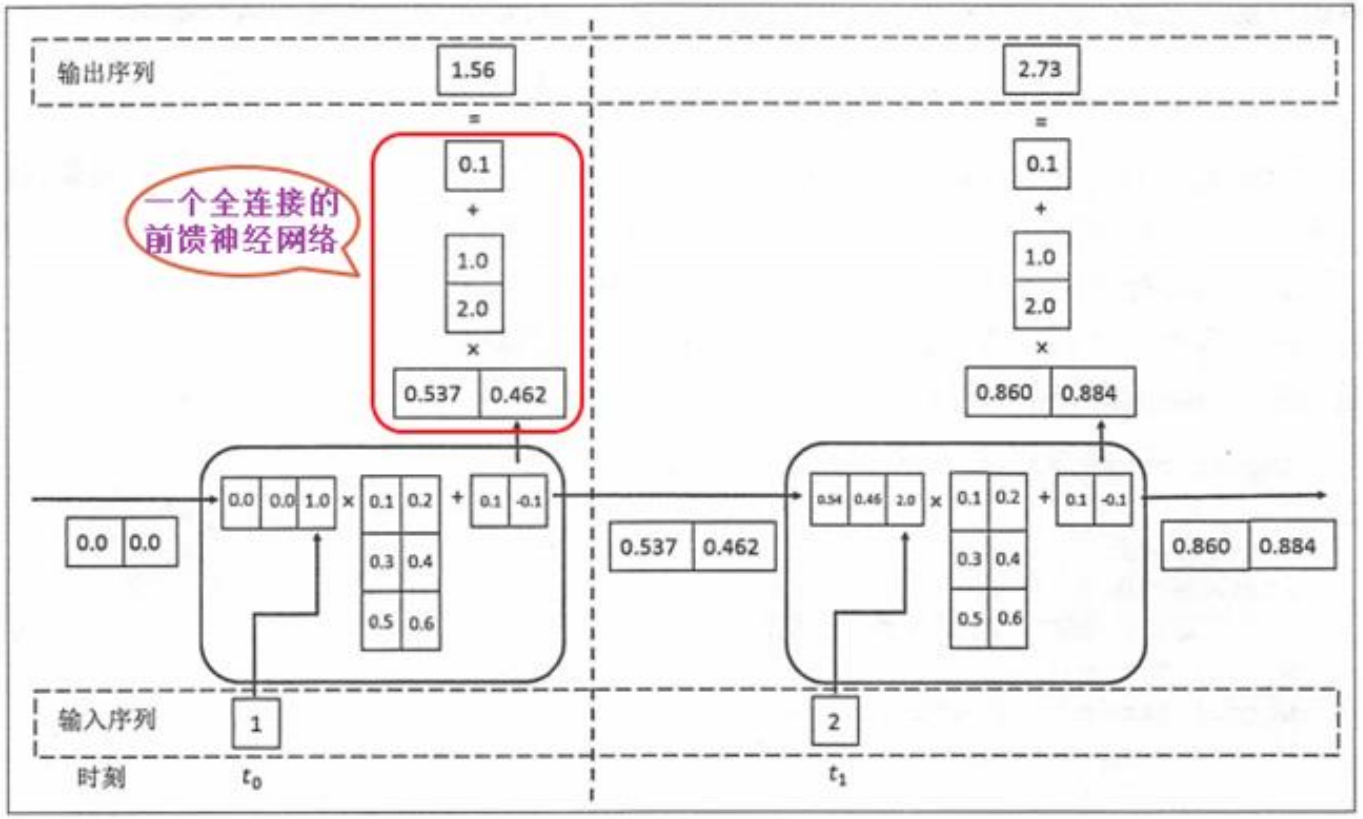
$$S_2 = w_x x_2 + w_s S_1 + b_1 \quad S_1 = \sigma(S_1) \quad O_2 = W_O S_2 + b_2$$

$$S_3 = w_x x_3 + w_s S_2 + b_1 \quad S_1 = \sigma(S_1) \quad O_3 = W_O S_3 + b_2$$

Loss损失函数 $Loss = \frac{1}{2}(Y_3 - O_3)^2$

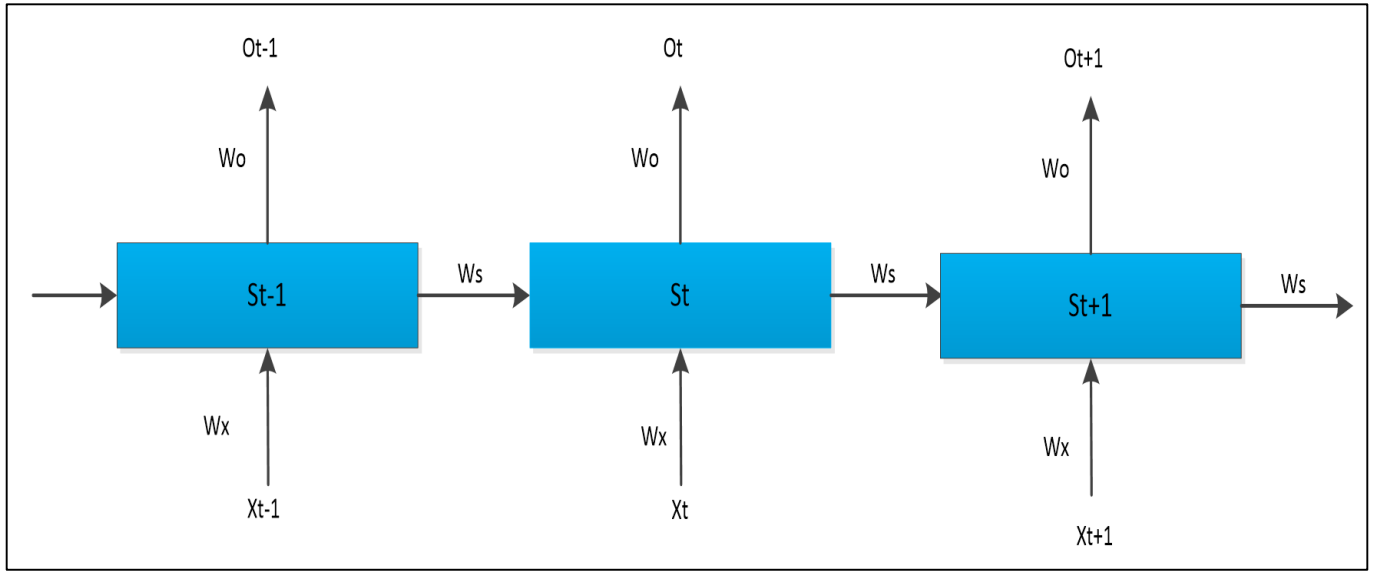
$\sigma = \tanh(x)$





02 RNN正向传播示意

03 RNN示意



04 RNN反向传播示意

$$S_1 = w_x x_1 + w_s S_0 + b_1 \qquad O_1 = W_O S_1 + b_2$$

$$S_2 = w_x x_2 + w_s S_1 + b_1 \qquad O_2 = W_O S_2 + b_2$$

$$S_3 = w_x x_3 + w_s S_2 + b_1 \qquad O_3 = W_O S_3 + b_2$$

Loss损失函数 $Loss = \frac{1}{2}(Y_3 - O_3)^2$

05 RNN反向传播示意-只对t3时刻进行反向传播

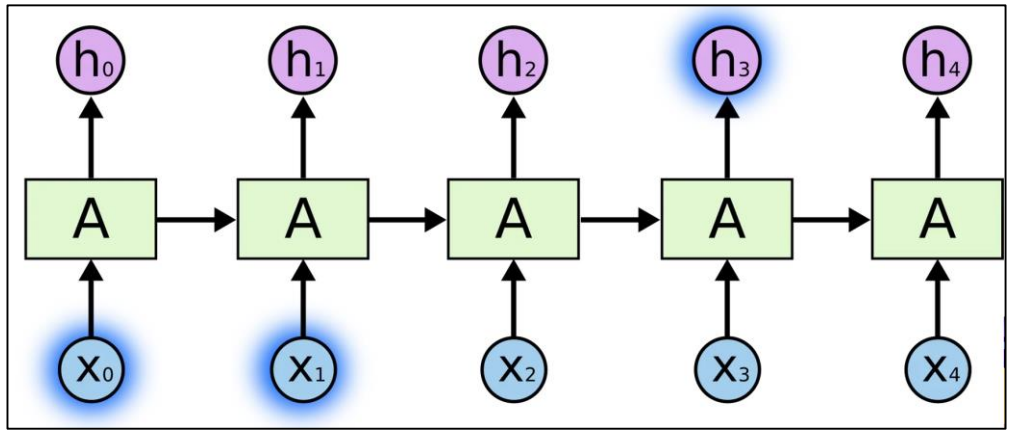
$$\frac{\partial L_3}{\partial W_0} = \frac{\partial L_3}{\partial O_3}\frac{\partial O_3}{\partial W_0} = (Y_3 - O_3)\cdot S_3$$

$$\frac{\partial L_3}{\partial Wx} = \frac{\partial L_3}{\partial O_3}\frac{\partial O_3}{\partial S_3}\frac{\partial S_3}{\partial W_x} + \frac{\partial L_3}{\partial O_3}\frac{\partial O_3}{\partial S_3}\frac{\partial S_3}{\partial S_2}\frac{\partial S_2}{\partial W_x} + \frac{\partial L_3}{\partial O_3}\frac{\partial O_3}{\partial S_3}\frac{\partial S_3}{\partial S_2}\frac{\partial S_2}{\partial S_1}\frac{\partial S_1}{\partial W_x}$$

$$\frac{\partial L_3}{\partial Ws} = \frac{\partial L_3}{\partial O_3}\frac{\partial O_3}{\partial S_3}\frac{\partial S_3}{\partial W_s} + \frac{\partial L_3}{\partial O_3}\frac{\partial O_3}{\partial S_3}\frac{\partial S_3}{\partial S_2}\frac{\partial S_2}{\partial W_s} + \frac{\partial L_3}{\partial O_3}\frac{\partial O_3}{\partial S_3}\frac{\partial S_3}{\partial S_2}\frac{\partial S_2}{\partial S_1}\frac{\partial S_1}{\partial W_s}$$

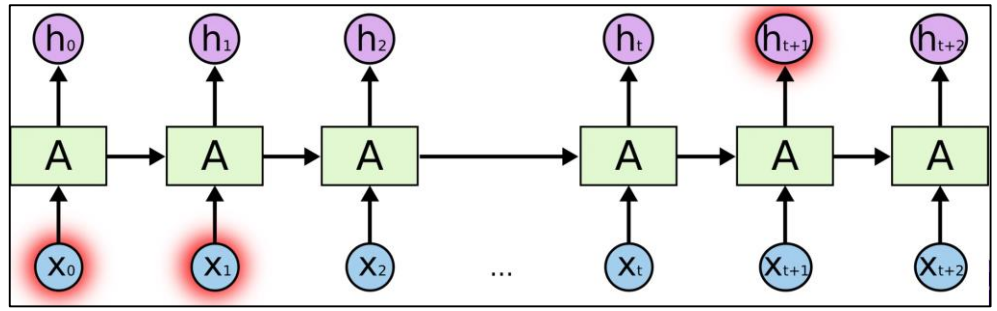
LSTM

01 RNN的缺点



02 RNN的缺点

序列数据广泛存在于各类任务中，如**中英文翻译**、**语音识别**等。这类任务通常需要实现**输入序列**到**输出序列**的-**转换梯度爆炸**



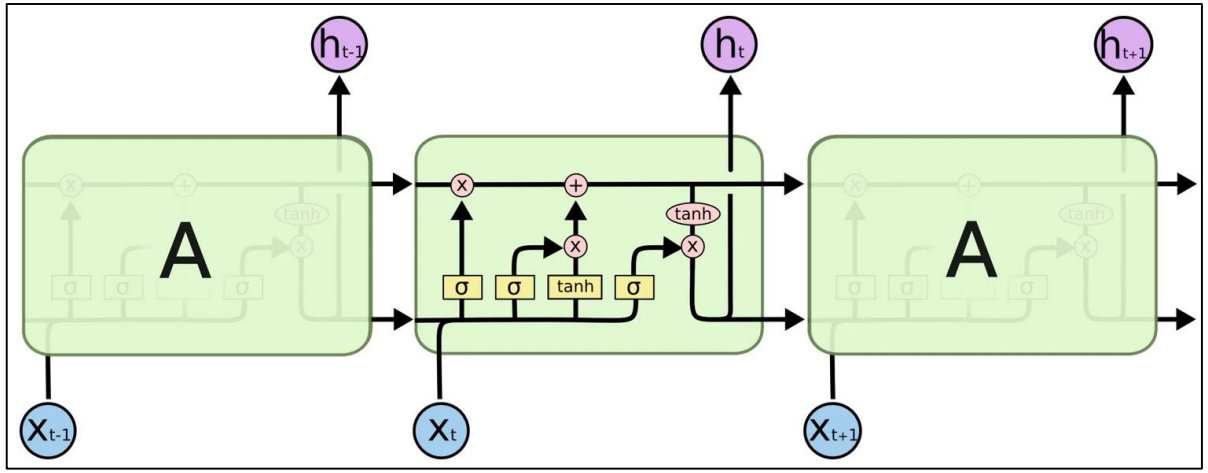
03 什么是lstm

Long Short Term长短期记忆

1）哪些细胞状态应该被遗忘

2）哪些新的状态应该被加入

3）根据当前的状态和现在的输入，输出应该是什么



遗忘门: 决定从细胞状态中丢弃什么信息,通过当前时刻输入和前一个时刻输出决定

细胞状态: 确定并更新新信息到当前时刻的细胞状态中

输出门: 基于目前的细胞状态决定该时刻的输出

04 样例

现有一个样本，样本维度为25*6

这样一共有25个时间步，每个时间步的特征长度是6.

我们现在要新增一个LSTM层，设置为

keras.layers.LSTM(10)，这样我们的中间层隐变量是10维

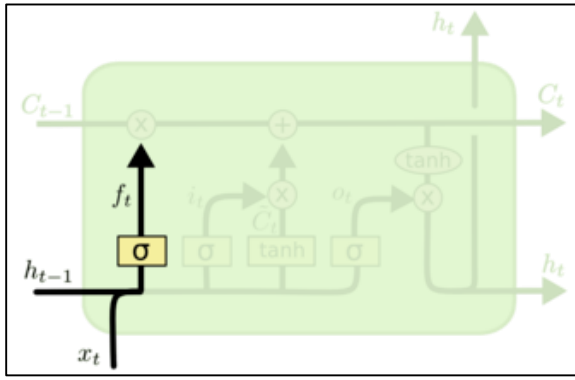
$$\begin{pmatrix} x_1^1 & x_1^2 \cdots x_1^6 \\ x_2^1 & x_2^2 \cdots x_2^6 \\ \vdots & \vdots \quad \vdots \\ x_{25}^1 & x_{25}^2 \cdots x_{25}^6 \end{pmatrix}$$

05 lstm-遗忘门

遗忘门: 决定从细胞状态中丢弃什么信息,通过当前时刻输入和前一个时刻输出决定。

h_{t-1}为上一个状态的隐向量（维度为10），x_t当前的状态特征长度为6，这样$[h_{t-1},x_t]$的维度为10+6=16维度，W_f和b_f为该层的参数.

$[h_{t-1},x_t]$为（1,16）维度的向量，与W_f矩阵相乘得到的（1,10）维度向量并且和（1,10）维度向量的偏置函数b_f相加。这样W_f的维度为（16,10）



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

05 lstm-遗忘门

遗忘门: 决定从细胞状态中丢弃什么信息,通过当前时刻输入和前一个时刻输出决定。

ht-1为上一个状态的隐向量（维度为10），xt当前的状态特征长度为6，这样[ht-1,xt]的维度为

10+6=16维度，Wf和bf为该层的参数.

[ht-1,xt]为（1,16）维度的向量，与Wf矩阵相乘得到的（1,10）维度向量并且和（1,10）维度向量的偏置函数bf相加。

这样Wf的维度为（16,10）

06 lstm-细胞状态

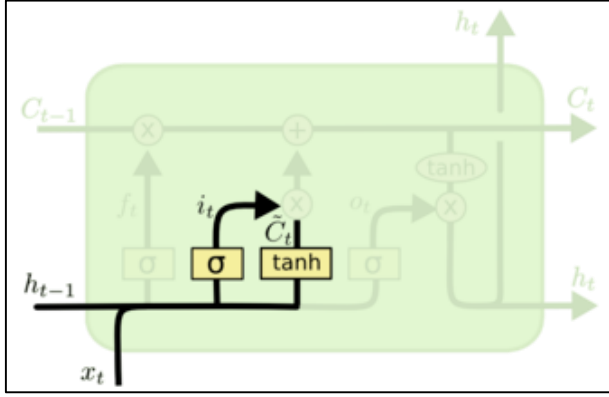
细胞状态: 确定并更新新信息到当前时刻的细胞状态中。

这样，细胞状态中

h_{t-1}为上一个状态的隐向量（维度为10），x_t当前的状态特征长度为6，这样$[h_{t-1},x_t]$的维度10+6=16维度，W_i和b_i为该层的参数.

$[h_{t-1},x_t]$为（1,16）维度的向量，与W_i矩阵相乘得到的（1,10）维度向量并且和（1,10）维度向量的偏置函数b_i相加。这样W_i的维度为（16,10）

$[h_{t-1},x_t]$为（1,16）维度的向量，与W_C矩阵相乘得到的（1,10）维度向量并且和（1,10）维度向量的偏置函数b_C相加。这样W_C的维度为（16,10）

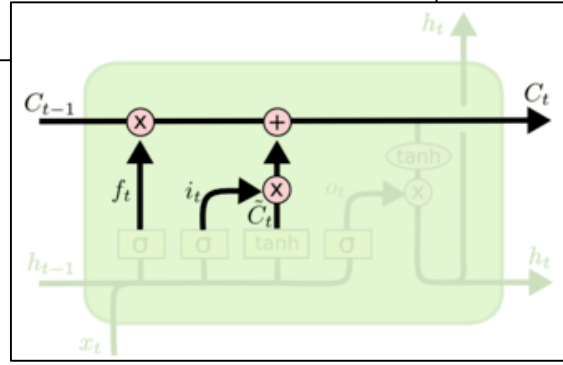


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

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

07 lstm-更新状态

细胞状态: 确定并更新新信息到当前时刻的细胞状态中。



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

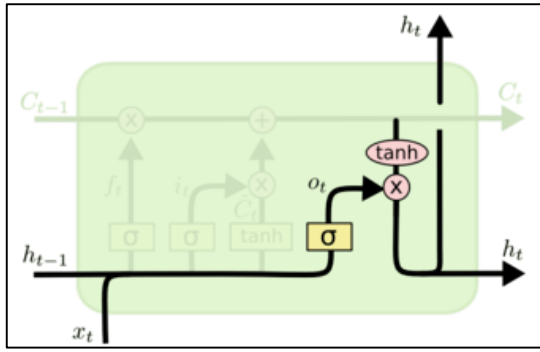
08 lstm-输出层

输出门: 基于目前的细胞状态决定该时刻的输出

这样，细胞状态中.

h_{t-1}为上一个状态的隐向量（维度为10），x_t当前的状态特征长度为6，这样$[h_{t-1},x_t]$的维度为10+6=16维度，W_i和b_i为该层的参数.

$[h_{t-1},x_t]$为（1,16）维度的向量，与W_o矩阵相乘得到的（1,10）维度向量并且和（1,10）维度向量的偏置函数b_o相加。这样W_o的维度为（16,10）



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

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

RNN-Keras实现
LSTM-Keras实现