RNN、LSTM、GRU

1. RNN

传统的RNN也就是BasicRNNcell单元。它的内部运算过程为，t-1时刻的隐层输出与w矩阵相乘，与t时刻的输入乘以u矩阵后进行相加，然后在经过一个非线性变化(tanh和Relu)，然后再以此方式作为隐层输出传给下一时刻。

RNN的结构如下图：

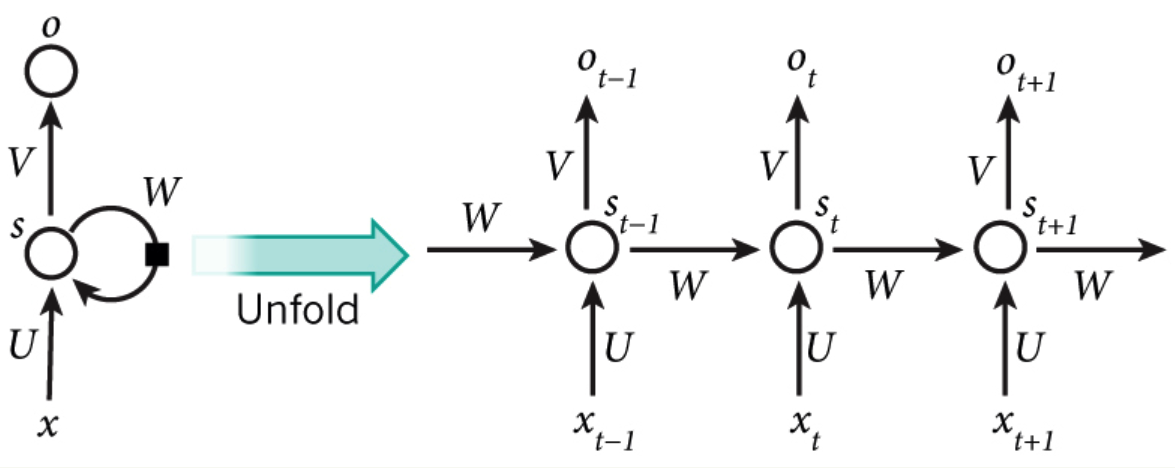


图1：RNN的结构

传统的RNN每一步的隐层单元都只是执行一个简单的tanh或者Relu操作。**在RNN里面，如果网络的层次太深的话，此时会产生梯度消失或者梯度爆炸问题。是因为RNN如果有多个时刻输入的时候，网络层次比较深，此时反向传播的路径比较长。反向传播是根据链式法则，如果开始的梯度小于1的话，到最后时刻的梯度几乎为0，则可以理解为梯度消失；反之，若开始的梯度大于1的话则最后时刻的梯度则非常大，可理解为梯度爆炸，这种情况，可以使用Relu函数。**

**为什么应用Relu函数呢？Relu函数在小于0时的梯度为0，大于0的时候梯度为1.使用Relu的好处就是：梯度容易求解；不会产生梯度消失或者梯度爆炸。**

1. LSTM

LSTM的结构及公式如下：

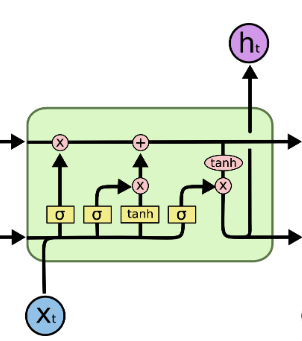


图2：LSTM的结构

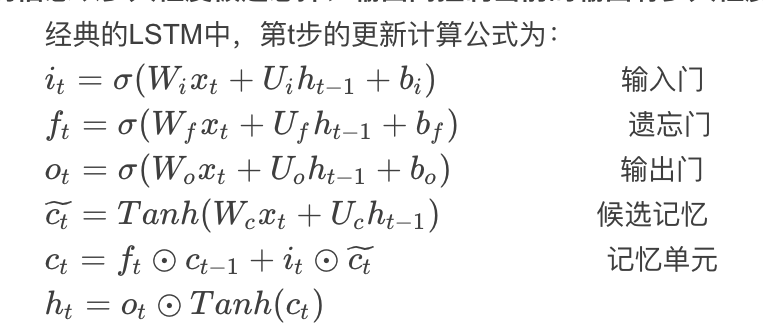


图3：LSTM的公式

**输入门控制当前计算的新状态以多大的程度更新到记忆单元中；遗忘门控制前一步记忆单元中的信息以多大的程度被遗忘掉；输出门控制当前的输出有多大程度取决于当前的记忆单元。**

**与传统的RNN不同的是，从上一个记忆单元到当前状态的转移不一定完全取决于激活函数计算得到的状态，还得由输入门和遗忘门共同控制。**

**在一个训练好的网络中，当输入序列没有重要信息时，LSTM的输入门接近于0，遗忘门的值接近于1，此时过去的记忆会被保存，从而实现了长期记忆；当输入的序列中出现了重要信息时, LSTM会将其存入记忆中，此时输入门的值接近于1；当输入序列中出现重要信息，且以前的记忆不再重要时，输入门接近于1，遗忘门接近于0.**

1. GRU

GRU在LSTM的基础上做了两个改进：

* 将输入门和遗忘门合并为一个门—更新门（update gate）。同时原来的输出门对应现在的重置门（reset gate）。
* 将单元状态（隐层状态）和输出门合并为一个状态：h

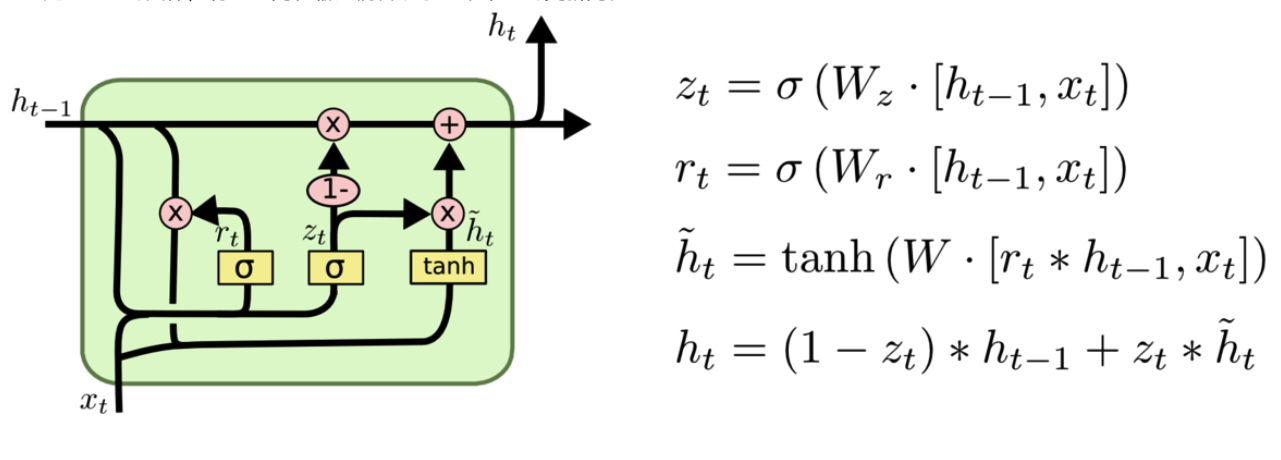


图4：GRU的结构