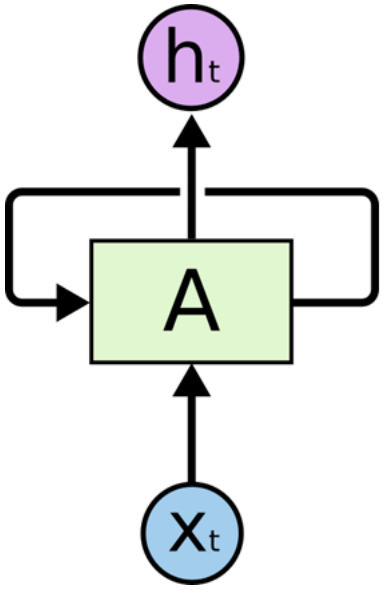
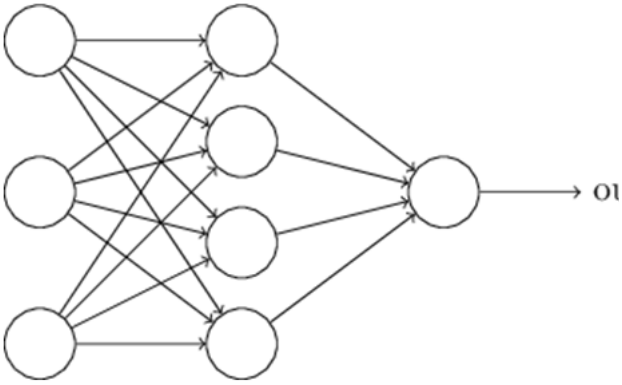
循环神经网络

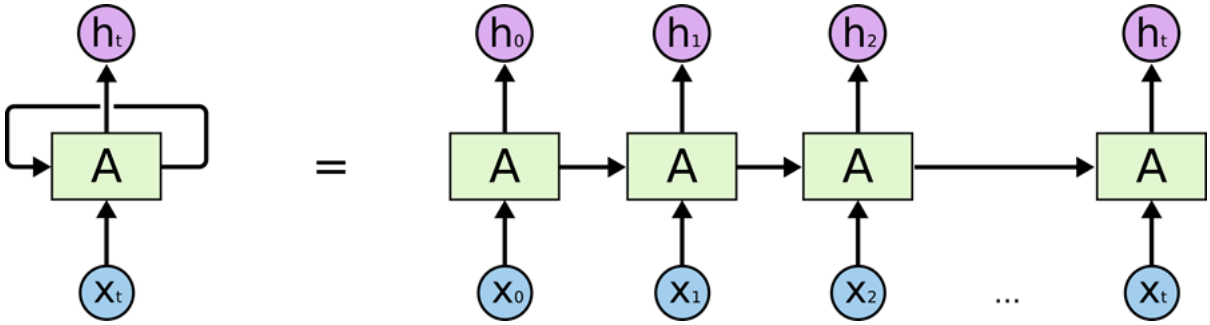
机器学习（入门） DC学院

循环神经网络

循环神经网络（recurrent neural network，即RNN）是一类用于处理时间序列数据的神经网络。传统的神经网络输入输出彼此独立，无法记忆相关信息。循环神经网络优势在于对序列的每个元素进行相同的计算，输出取决于之前的数据状态，可以利用任意序列中的信息。

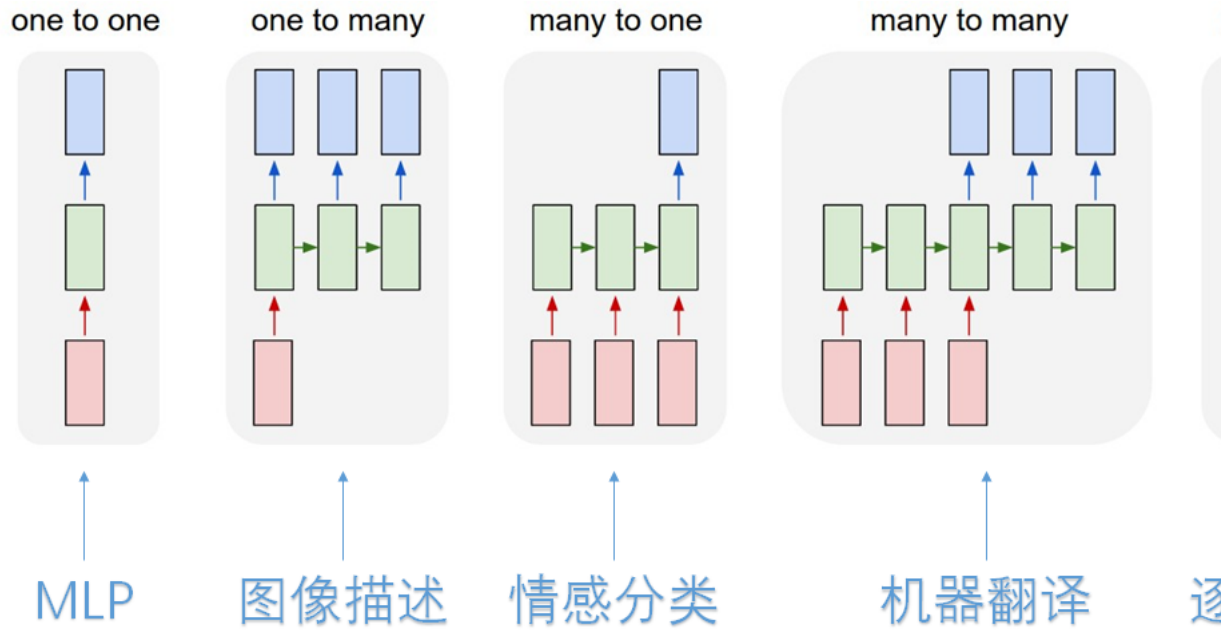






- X_t–在t时刻的输入数据
- s_t–在t时刻RNN的隐藏状态（hidden state）
- h_t–在t时刻RNN对应的输出数据
- U–输入相关的权重矩阵
- W–隐藏状态相关的权重矩阵
- V–输出相关的权重矩阵

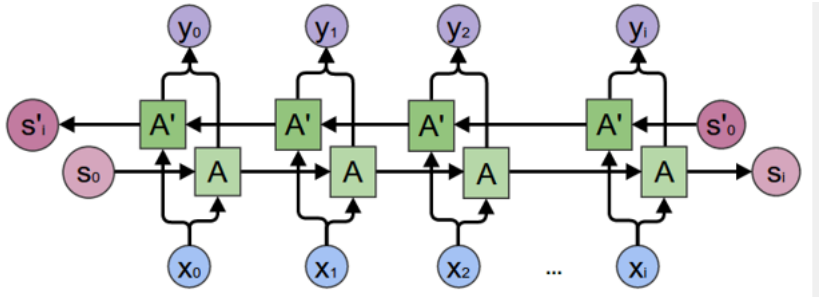
$$s_t = f(UX_t + Ws_{t-1}) \qquad h_t = softmax(Vs_t)$$



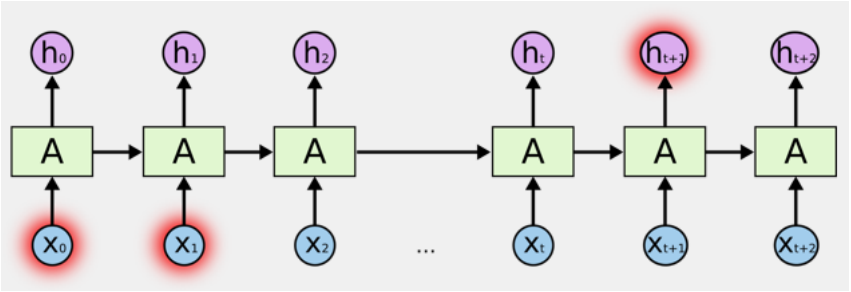
循环神经网络一个很重要的问题在于网络只能从之前时刻的输入中进行学习，而序列中某个时刻的信息可能是和上下文相关的。

双向RNN

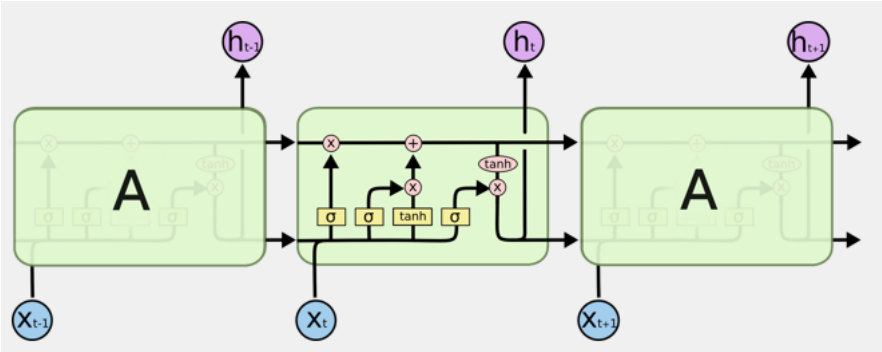
双向循环神经网络（Bidirectional RNN）的基本思想是提出每一个训练序列向前和向后分别是两个循环神经网络，而且这两个都连接着一个输出层。这个结构提供给输出层输入序列中每一个点的完整的过去和未来的上下文信息。下图展示的是一个沿着时间展开的双向循环神经网络。



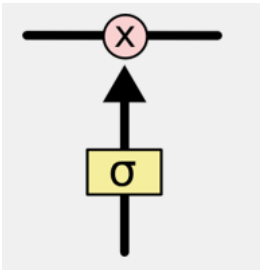
标准循环神经网络的另外一个问题在于能够存取的上下文信息范围很有限，隐含层的输入对于网络输出的影响随着时间的**不断递归而衰退**。



长短期记忆网络（Long Short-Term Memory，LSTM）是一种特殊的RNN结构，通过对结构的设计可以有效的处理RNN在长期依赖上存在的问题。

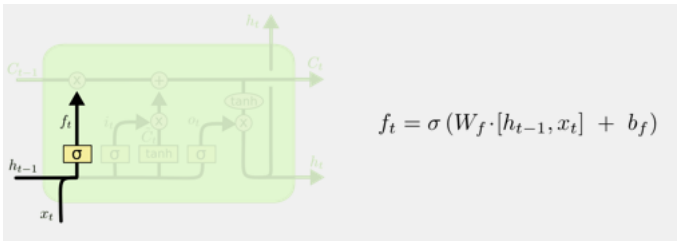


LSTM中最基本的就是“门”（gate）的结构，让信息有选择性的影响网络中每个时刻的状态。“门”结构就是使用sigmoid神经网络和一个按照元素相乘的操作。



遗忘门

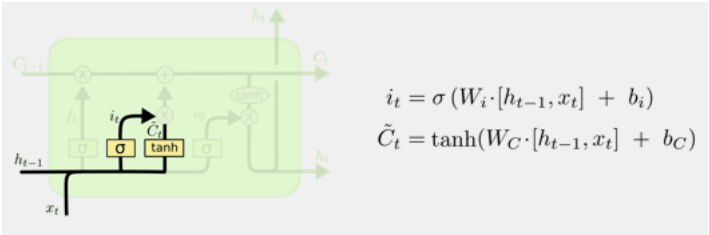
“遗忘门”和“输入门”是LSTM结构的核心，“遗忘门”的作用是让LSTM“忘记”之前没有用的信息。



- C_t $-t$时刻的隐藏状态
- h_t $-t$时刻的输出
- x_t $-t$时刻的输入

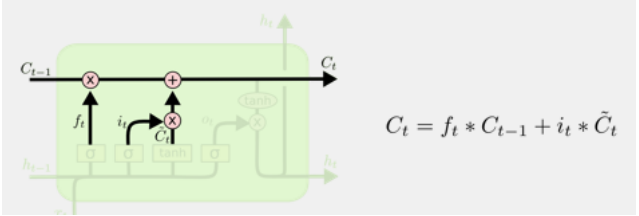
输入门

LSTM忘记了之前的部分信息后还需要从当前的输入数据中补充新的数据信息，这样的一个过程就是“输入门”完成的。



- C_t $-t$时刻的隐藏状态
- h_t $-t$时刻的输出
- x_t $-t$时刻的输入

更新状态

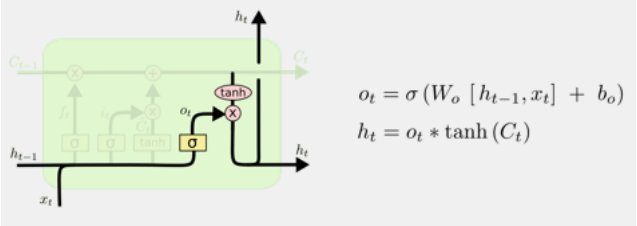


- C_t $-t$时刻的隐藏状态

- h_t–t时刻的输出
- x_t–t时刻的输入

输出门

LSTM在计算得到最新的隐藏状态后需要产生当前时刻的输出，这个过程是通过输出门来完成的。

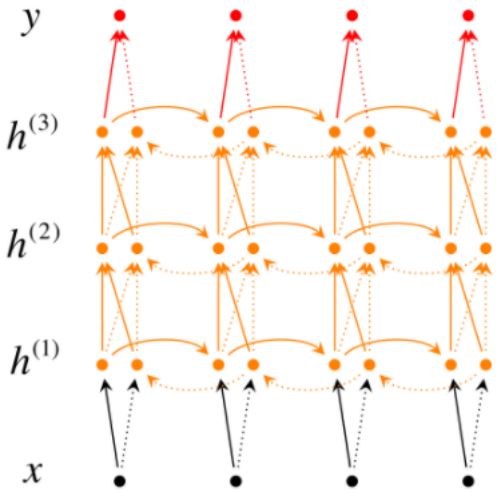


- C_t–t时刻的隐藏状态
- h_t–t时刻的输出
- x_t–t时刻的输入

补充知识

RNN的改进：深层双向RNN

深层双向RNN 与双向RNN相比，多了几个隐藏层，就是基于这么一个想法，他的输入有两方面，第一就是前一时刻的隐藏层传过来的信息 $\overrightarrow{h}_{t-1}^{(i)}$，和当前时刻上一隐藏层传过来的信息 $h_t^{(i-1)} = [\overrightarrow{h}_t^{(i-1)}; \overleftarrow{h}_t^{(i-1)}]$，包括前向和后向的。



用公式来表示是这样的:

$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}^{(i)} h_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)})$$

$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

优化长期依赖–截断梯度

梯度告诉我们，围绕当前参数的无穷小区域内最速下降的方向。
截断梯度(clipping the gradient)。

- 在参数更新之前，逐元素地截断小批量产生的参数梯度。
- 参数更新之前截断梯度g的范数||g||

$$if ||g|| > v$$
$$g \leftarrow \frac{gv}{||g||}$$

其中v是范数上届，g用来更新参数。因为所有参数的梯度被单个因子联合重整化，所以后一方法的优点是保证了每个步骤仍然是在梯度方向上的。

截断每小批量梯度范数不会改变单个小批量的梯度方向。然而，许多小批量使用范数截断梯度后的平均值不等于截断真实梯度（使用所有的实例所形成的梯度）的范数。

RNN优化算法-BPTT

BPTT 是求解RNN问题的一种优化算法，也是基于BP算法改进得到和BP算法比较类似，分为三步：

- 前向计算每个神经元的输出值；
- 反向计算每个神经元的误差项值，它是误差函数E对神经元j的加权输入的偏导数；
- 计算每个权重的梯度。

最后再用随机梯度下降算法更新权重。