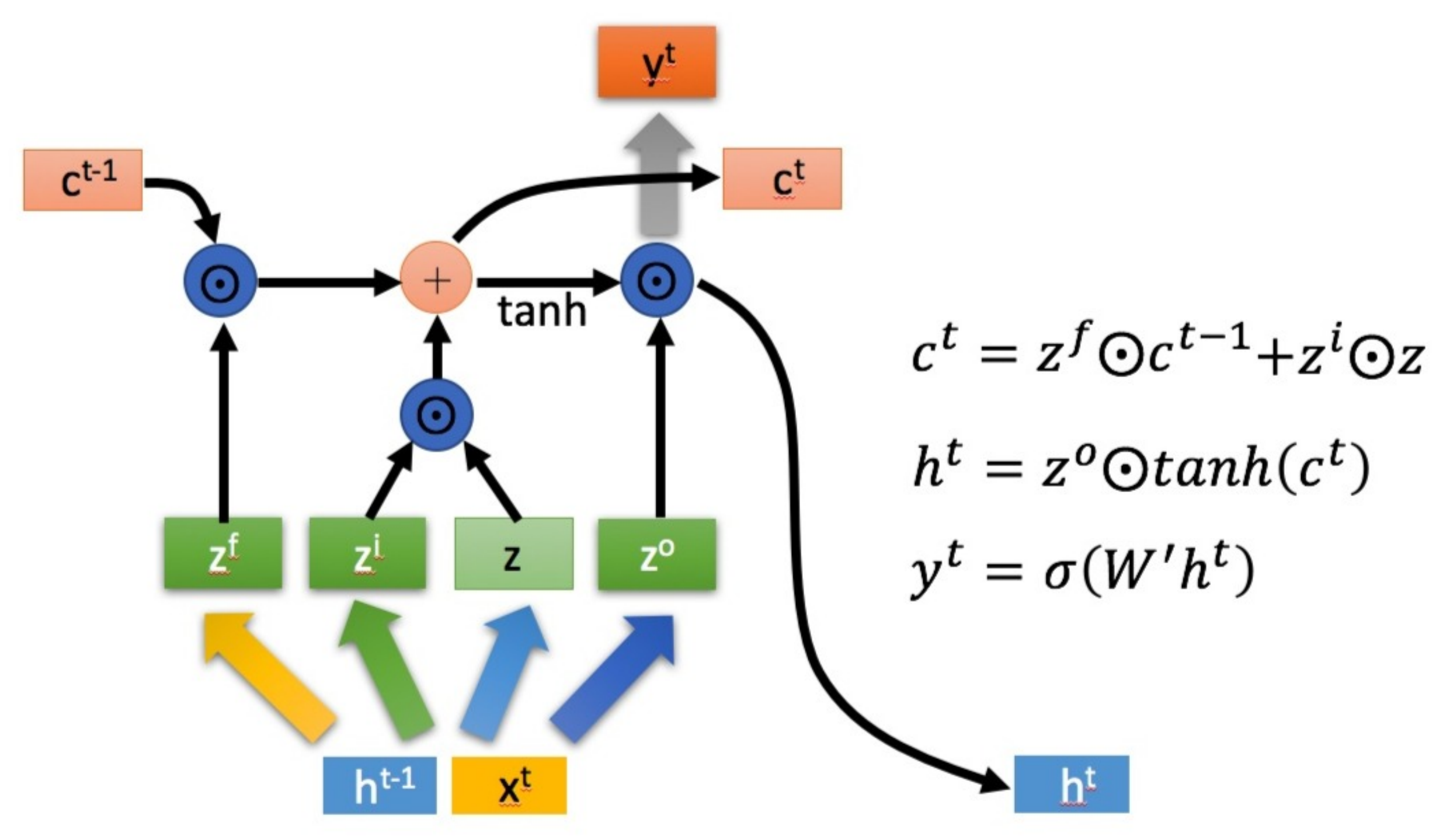


人人都能看懂的LSTM

陈诚

2,913 人赞同了该文章

> 这是在看了台大李宏毅教授的深度学习视频之后的一点总结和感想。看完介绍的第一部分RNN尤其LSTM的介绍之后，整个人醍醐灌顶。本篇博客就是对视频的一些记录加上了一些个人的思考。

---

0. 从RNN说起

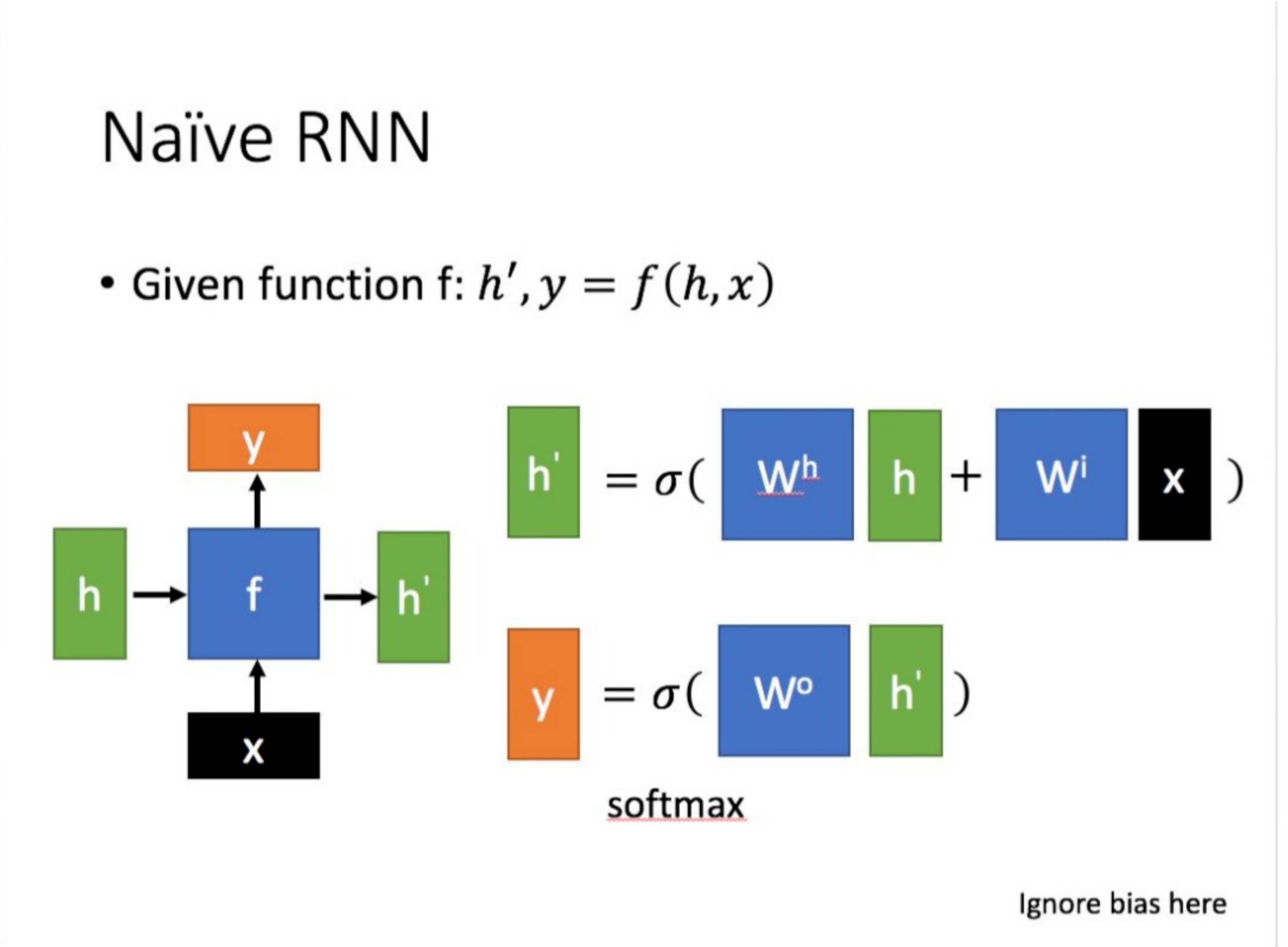
循环神经网络（Recurrent Neural Network，RNN）是一种用于处理序列数据的神经网络。相比一般的神经网络来说，他能够处理序列变化的数据。比如某个单词的意思会因为上文提到的内容不同而有不同的含义，RNN就能够很好地解决这类问题。

---

1. 普通RNN

先简单介绍一下一般的RNN。

其主要形式如下图所示（图片均来自台大李宏毅教授的PPT）：



这里：

x 为当前状态下数据的输入， h 表示接收到的上一个节点的输入。

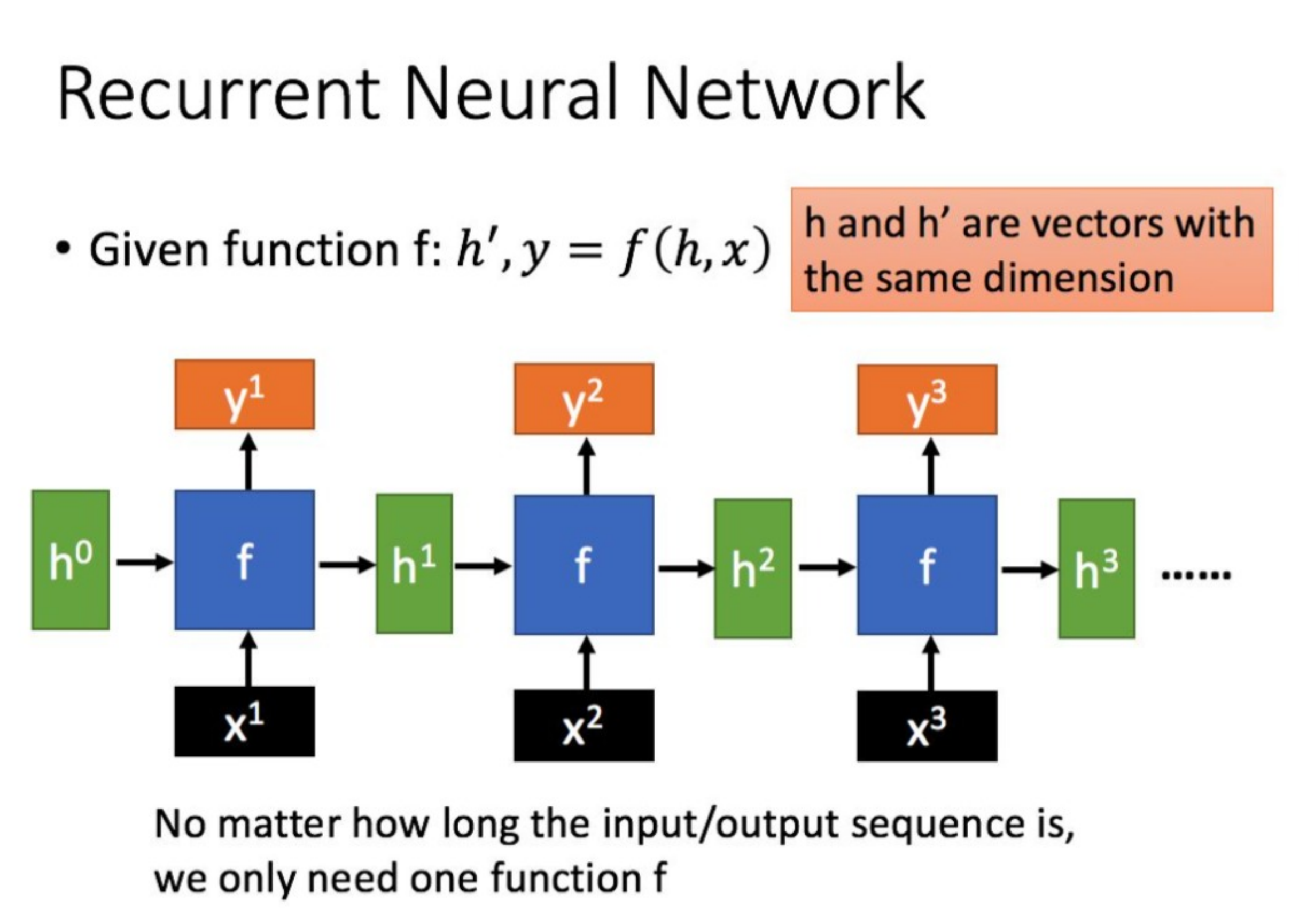
y 为当前节点状态下的输出，而 h' 为传递到下一个节点的输出。

通过上图的公式可以看到，输出 **h'** 与 **x** 和 **h** 的值都相关。

而 **y** 则常常使用 **h'** 投入到一个线性层（主要是进行维度映射）然后使用softmax进行分类得到需要的数据。

对这里的**y**如何通过 **h'** 计算得到往往看具体模型的使用方式。

通过序列形式的输入，我们能够得到如下形式的RNN。



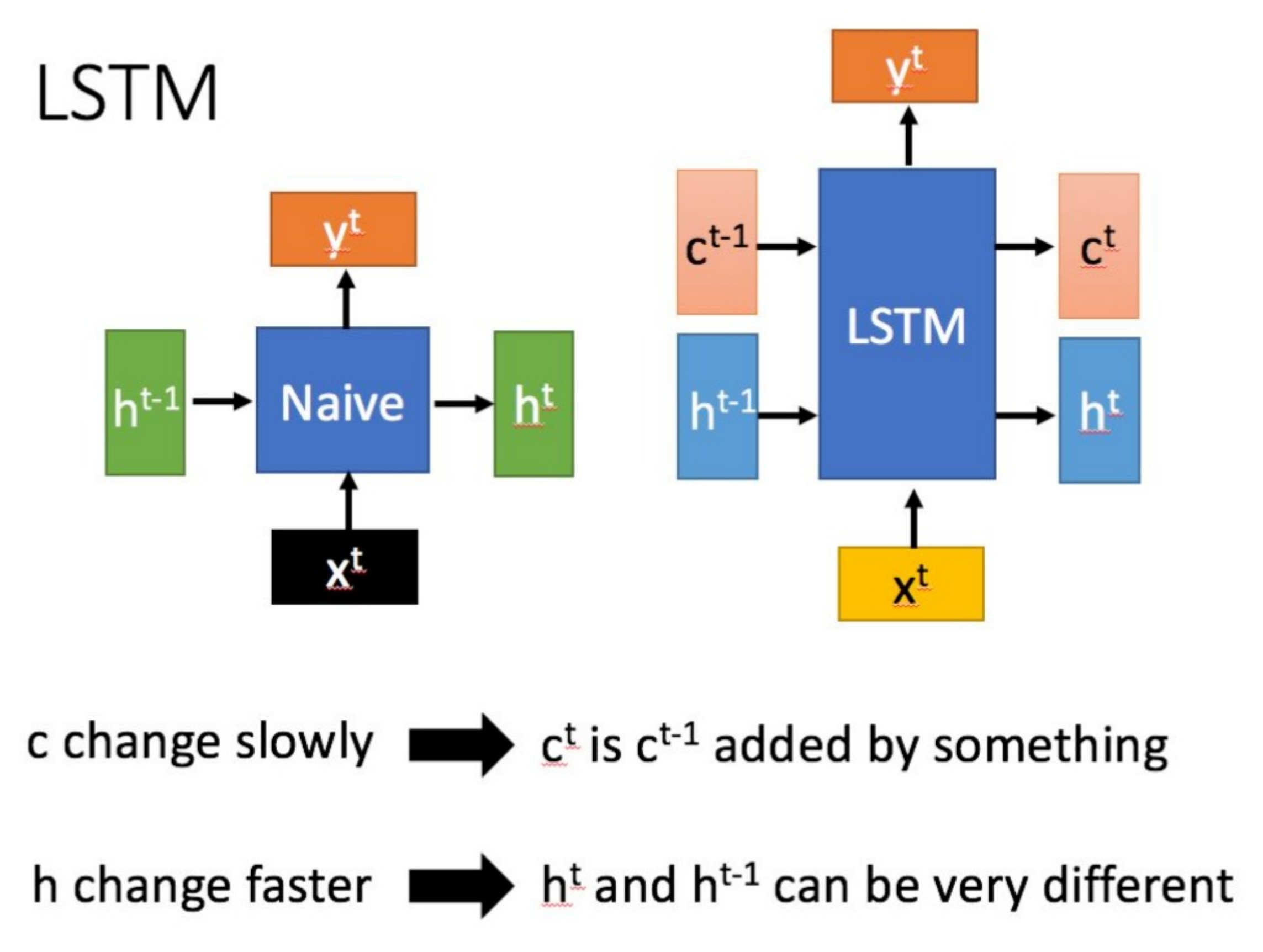
---

2. LSTM

2.1 什么是LSTM

长短期记忆（Long short-term memory, LSTM）是一种特殊的RNN，主要是为了解决长序列训练过程中的梯度消失和梯度爆炸问题。简单来说，就是相比普通的RNN，LSTM能够在更长的序列中有更好的表现。

LSTM结构（图右）和普通RNN的主要输入输出区别如下所示。



相比RNN只有一个传递状态 h^t ，LSTM有两个传输状态，一个 c^t （cell state），和一个 h^t （hidden state）。（Tips：RNN中的 h^t 对于LSTM中的 c^t ）

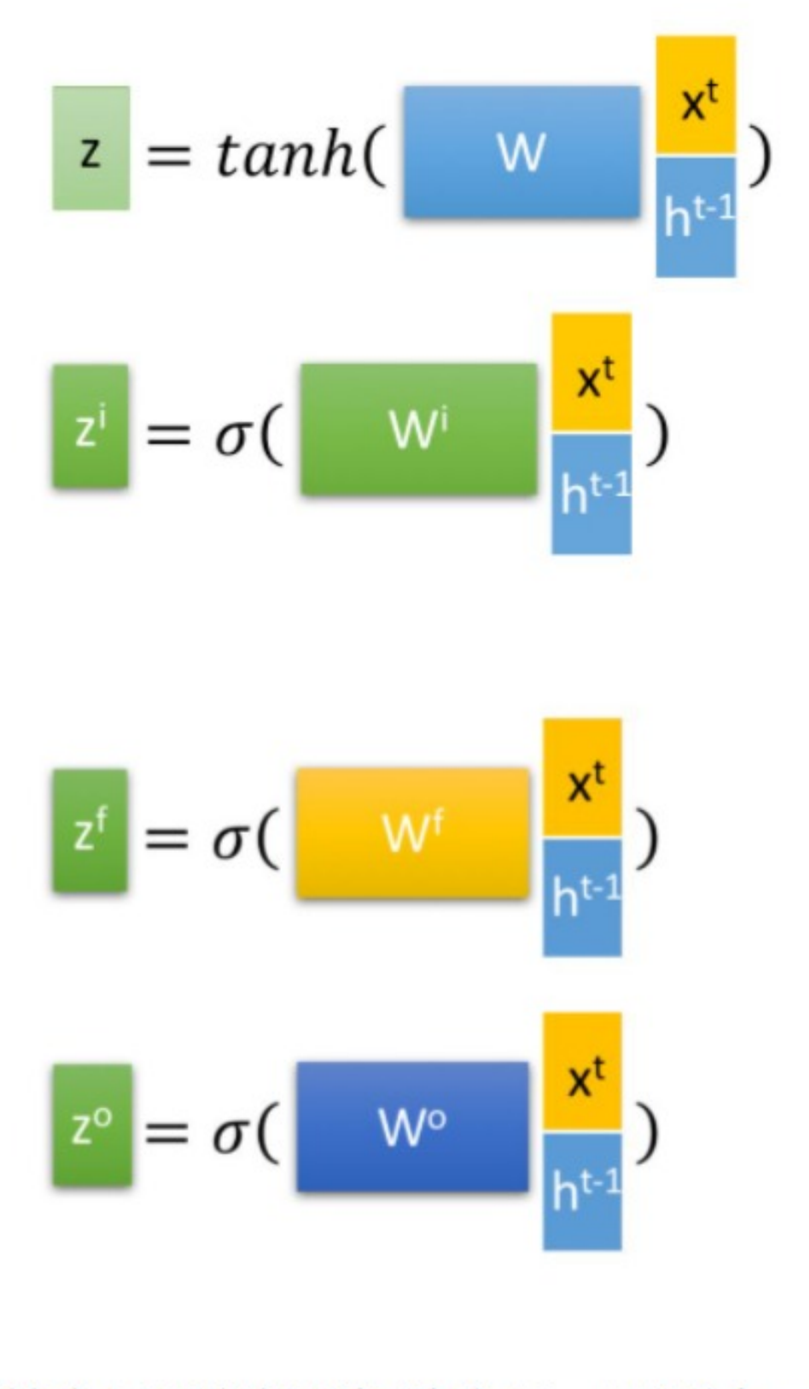
其中对于传递下去的 c^t 改变得很慢，通常输出的 c^t 是上一个状态传过来的 c^{t-1} 加上一些数值。

而 h^t 则在不同节点下往往会有很大的区别。

2.2 深入LSTM结构

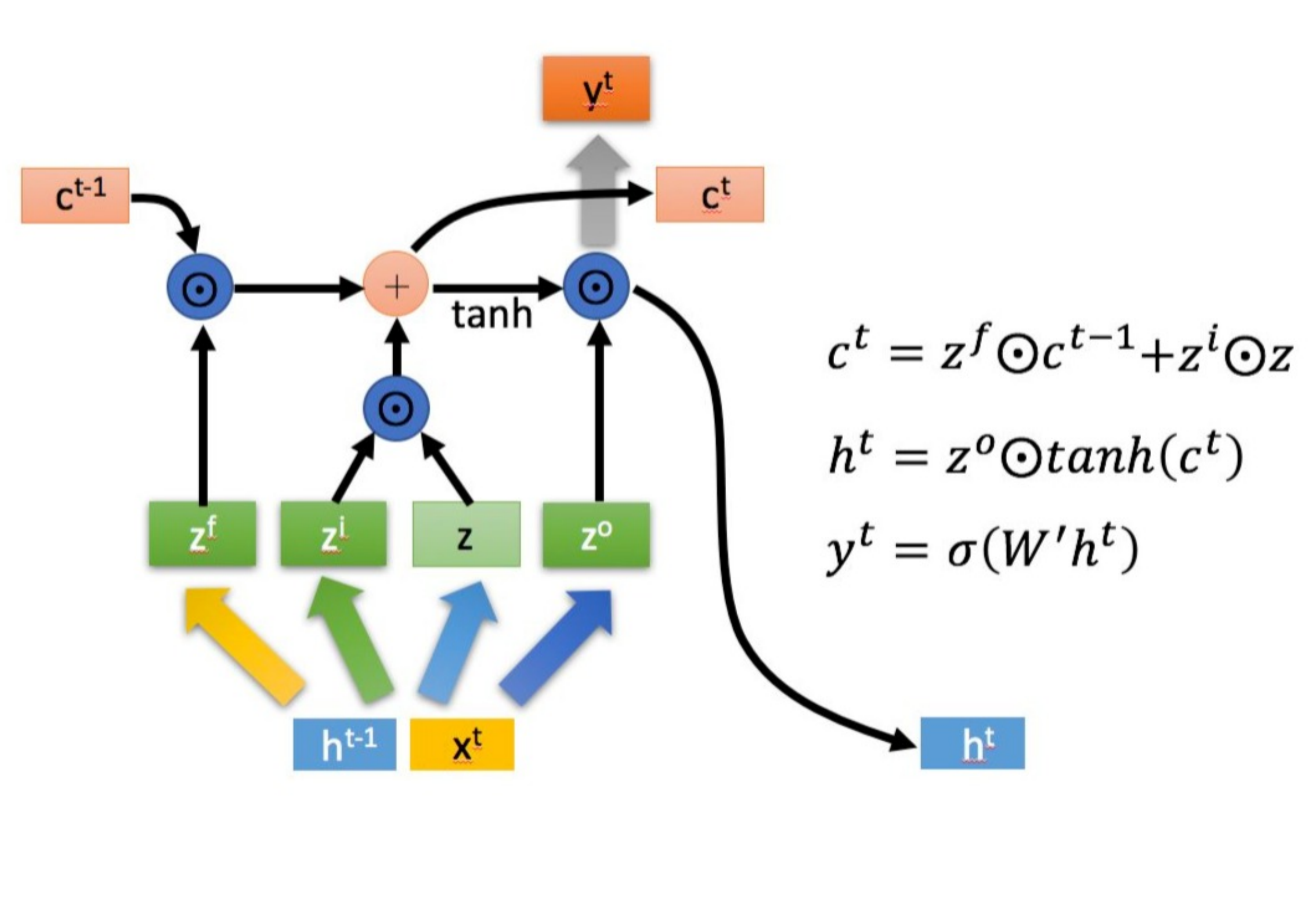
下面具体对LSTM的内部结构来进行剖析。

首先使用LSTM的当前输入 x^t 和上一个状态传递下来的 h^{t-1} 拼接训练得到四个状态。



其中， z^f ， z^i ， z^o 是由拼接向量乘以权重矩阵之后，再通过一个 $sigmoid$ 激活函数转换成0到1之间的数值，来作为一种门控状态。而 z 则是将结果通过一个 $tanh$ 激活函数将转换成-1到1之间的值（这里使用 $tanh$ 是因为这里是将其做为输入数据，而不是门控信号）。

下面开始进一步介绍这四个状态在LSTM内部的使用。（敲黑板）



$\odot$ 是Hadamard Product，也就是操作矩阵中对应的元素相乘，因此要求两个相乘矩阵是同型的。 $\oplus$ 则代表进行矩阵加法。

LSTM内部主要有三个阶段：

1. 忘记阶段。这个阶段主要是对上一个节点传进来的输入进行**选择性**忘记。简单来说就是会 “忘记不重要的，记住重要的”。

具体来说是通过计算得到的 z^f （f表示forget）来作为忘记门控，来控制上一个状态的 c^{t-1} 哪些需要留哪些需要忘。

2. 选择记忆阶段。这个阶段将这个阶段的输入有选择性地进行“记忆”。主要是会对输入 x^t 进行选择记忆。哪些重要则着重记录下来，哪些不重要，则少记一些。当前的输入内容由前面计算得到的 z 表示。而选择的门控信号则是由 z^i （i代表information）来进行控制。

> 将上面两步得到的结果相加，即可得到传输给下一个状态的 c^t 。也就是上图中的第一个公式。

3. 输出阶段。这个阶段将决定哪些将会被当成当前状态的输出。主要是通过 z^o 来进行控制的。并且还对上一阶段得到的 c^o 进行了放缩（通过一个tanh激活函数进行变化）。

与普通RNN类似，输出 y^t 往往最终也是通过 h^t 变化得到。

---

3. 总结

以上，就是LSTM的内部结构。通过门控状态来控制传输状态，记住需要长时间记忆的，忘记不重要的信息；而不像普通的RNN那样只能够“呆萌”地仅有一种记忆叠加方式。对很多需要“长期记忆”的任务来说，尤其好用。