问题

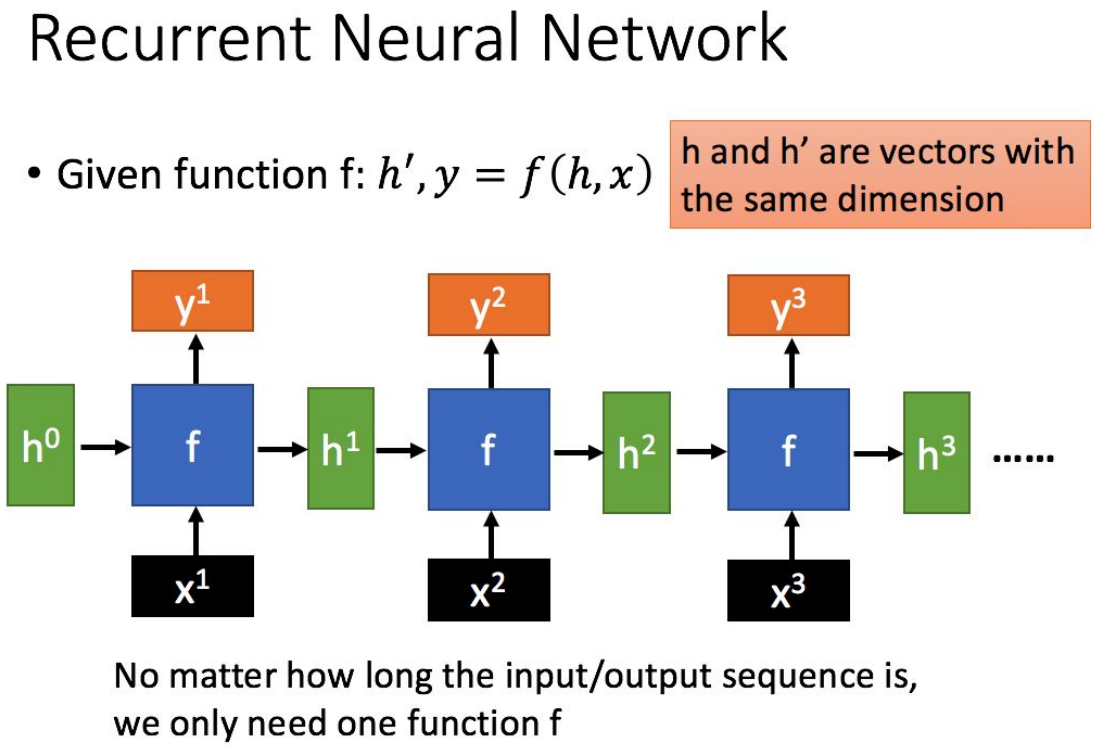
LSTM是深度学习语音领域必须掌握的一个概念，久仰大名，现在终于要来学习它了，真是世事无常，之前以为永远不会接触到呢，因此每次碰到这个就跳过了。

前言

LSTM (Long short-term memory，长短期记忆) 是一种特殊的RNN，主要是为了解决长序列训练过程中梯度消失与梯度爆炸的问题，因此要学习LSTM，必须先了解RNN是一个什么东东。

RNN

RNN (Recurrent Neural Network，循环卷积网络) 是一种用于处理序列数据的神经网络，比如文本分析中，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种问题。



我们以上图的第一个基本单元进行分析。

x^1 为当前状态下数据的输入，h^0 表示接收到的上一个节点的输入。
y^1 为当前节点状态下的输出，而 h^1 为传递到下一个节点的输出。
f 为一种映射函数，具体函数形式要看怎么设计。

h^1 和 y^1 的计算方式为：

$$\begin{aligned} h^1 &= \sigma(W^h h^0 + W^i x^1) \\ y^1 &= \sigma(W^o h^1) \end{aligned} \tag{1}$$

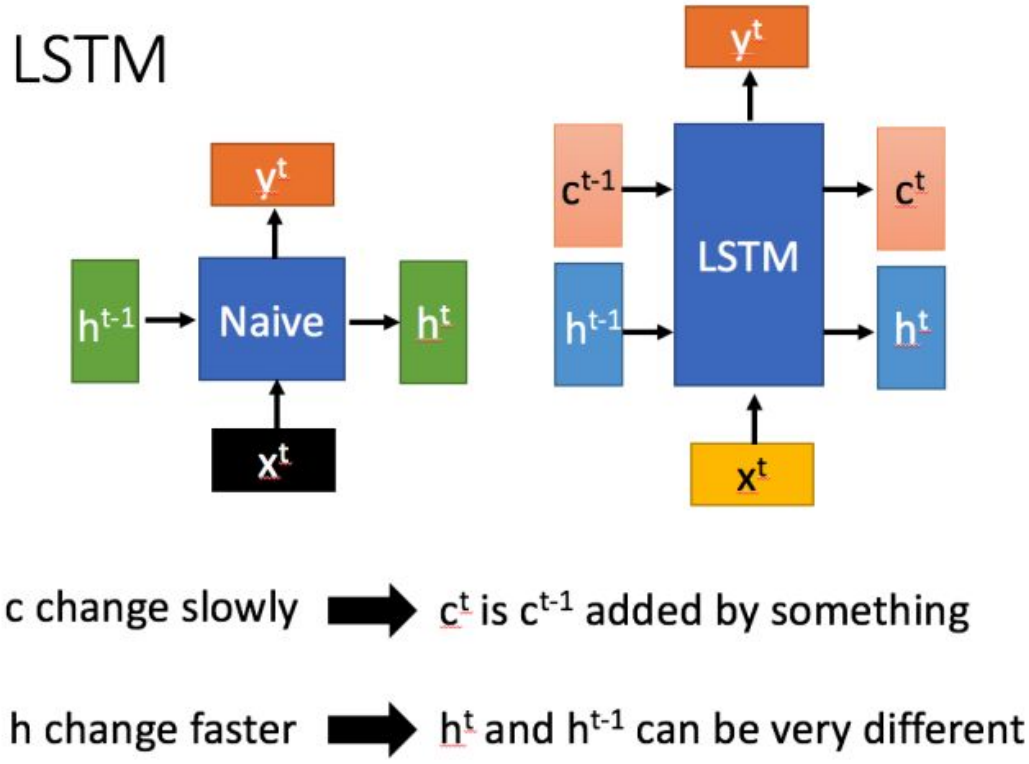
其中 W^h 、W^i 、W^o 为函数 f 的参数。

从图例和公式都可以看出，当前结点的输出不仅与当前结点相关，还与之前的所有结点相关。得到 $y^1, y^2, \ldots, y^n$ 之后，使用 softmax 函数便可得到所需的信息进行分类。（这里的输入 $x^1, x^2, \ldots, x^n$ 应该指的是每个字词单元的特征吧）

LSTM

介绍完RNN，下面开始介绍正主。如上所说，LSTM是在RNN的基础上改进而来的，解决的是长序列数据训练过程中的梯度消失和梯度爆炸问题。

LSTM 结构和普通 RNN 的主要输入输出区别如下图所示：

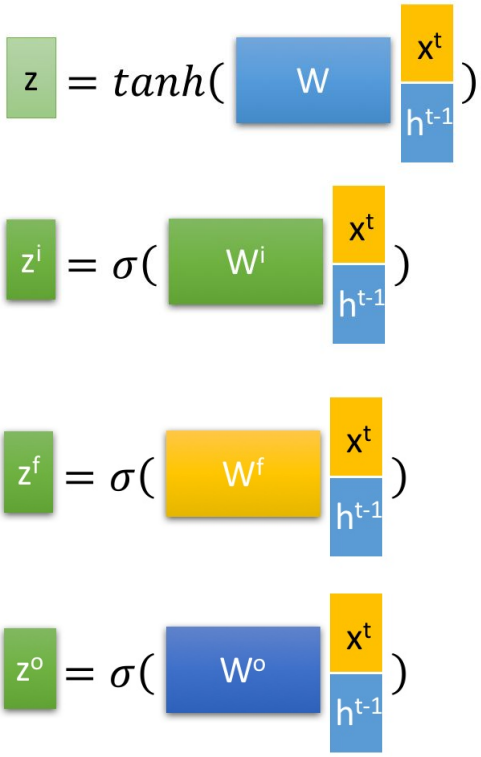


即 LSTM 结构相比于普通的 RNN 多了一个传输状态 c^t (cell state)， h^t 为 hidden state。参考资料中说RNN中的 h^t 对应于 LSTM 中的 c^t，传递过程中改变较慢。

> c^t 也就是cell state中的内容，可以理解为主线，主要是用来保存节点传递下来的数据的，每次传递会对某些维度进行“忘记”并且会加入当前节点所包含的内容，总的来说还是用来保存节点的信息，改变相对较小。而h^t 则主要是为了和当前输入组合来获得门控信号，对于不同的当前输入，传递给下一个状态的h^t区别也会较大。
>
> 上面说到LSTM中的 c^t 主要是用来保存先前节点的数据的，那么RNN只有 h^t，那么这个 h^t 肯定主要是保存了先前节点的信息的，所以我们说RNN中的 h^t 实际上对应的是LSTM中的 c^t。

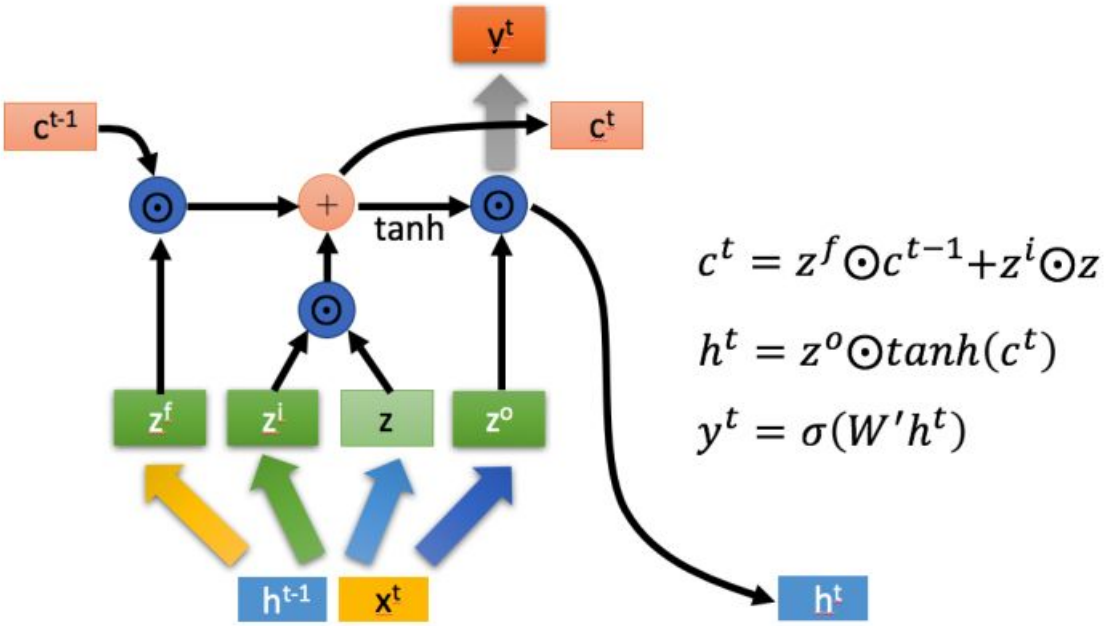
深度解读 LSTM 结构

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} 拼接得到四个状态。



其中，z^i、z^f 、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 激活函数将其转换为 -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）

这四个状态在 LSTM 内部是怎么使用的呢？（把这张图印在脑子里吧）



其中 $\odot$ 运算符表示矩阵中对应的元素相乘，而 $\oplus$ 表示进行矩阵加法。因此要求两个矩阵是同型的。

上图表达了 LSTM 的三个阶段：

1. **忘记阶段**：这个阶段主要是对上一个结点传进来的输入进行**选择性**忘记，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 代表 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记。
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 。
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^t 进行了放缩（通过一个 tanh 激活函数进行变化）。

与普通 RNN 类似，输出的 y^t 最终也是通过 h^t 变化得到。

这里再摘抄一下评论中对 c^t 和 h^t 的理解：

> 无论是 RNN 还是 LSTM ，h^t 感觉表示的都是短期记忆，RNN 相当于LSTM 中的最后一个“输出门”的操作，是 LSTM 的一个特例，也就是 LSTM 中的短期记忆知识，而 LSTM 包含了长短期的记忆，其中 c^t 就是对前期记忆的不断加工，锤炼和理解，沉淀下来的，而 h^t 只是对前期知识点的短暂记忆，是会不断消失的。
>
> c^t 之所以变化慢，主要是对前期记忆和当前输入的线性变换，对前期记忆的更新和变化（可以视为理解或者领悟出来的内容），是线性变换，所以变动不大，而 h^t 是做的非线性变化，根据输入节点内容和非线性变化函数的不同，变动固然很大

LSTM 解决了需要长期记忆的任务的问题，比较好用，但也因为引入了很多内容，导致参数变多，也使得训练难度加大了很多。因此很多时候我们往往会使用效果和LSTM相当但参数更少的GRU来构建大训练量的模型。

参考资料

[人人都能看懂的LSTM]