循环神经网络介绍

前面我们介绍了非常适合处理图像的卷积神经网络，在深度学习的领域，除了视觉的问题，还有很多关于文本，语音的问题，这些问题不同于视觉这种结构性的问题，而是需要时间依赖和记忆性，这个时候使用卷积神经网络就不再那么适合，所以需要使用一种新的形式的网络，循环神经网络。

问题介绍

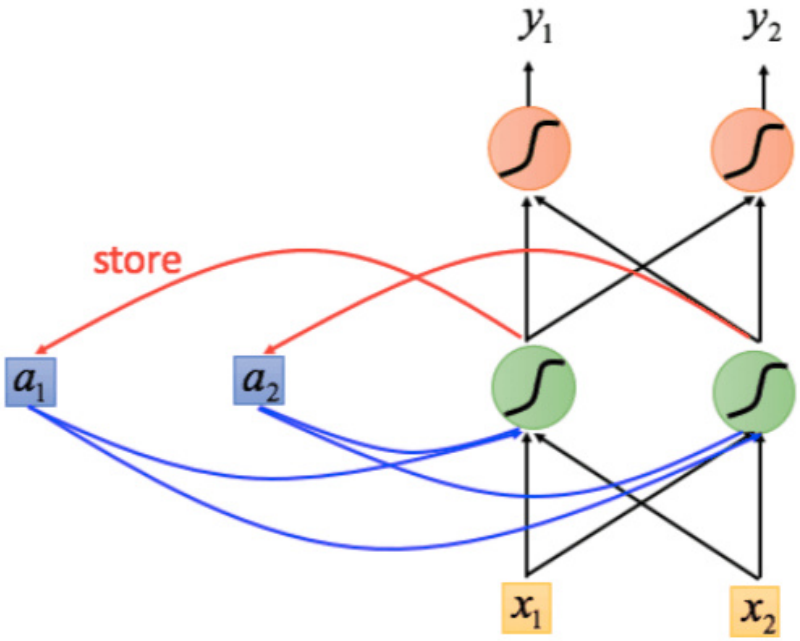
我们来举一个具体的例子引出循环神经网络。

我们来看看下面这两段话

第一句话表达了到达北京的意思，第二句话表达的是离开北京的意思，对于这样一个任务，如果网络只输入是 'beijing'，那么对于这两句话，网络会输出相同的结果，但是这是不符合我们直觉的，两句话应该表达不同的意思，那么有没有办法能够达到这个效果呢？其实这两句话中只有一个词的差别，就是'离开'和'到达'，如果我们能够用这两个词的信息就能够得到不同的结果，但是现在我们只有 'beijing' 这个输入，所以如果网络能够记忆 'beijing' 前面这个词，他就会预测出不同的结果，所以说这种任务需要神经网络具有记忆的特性。

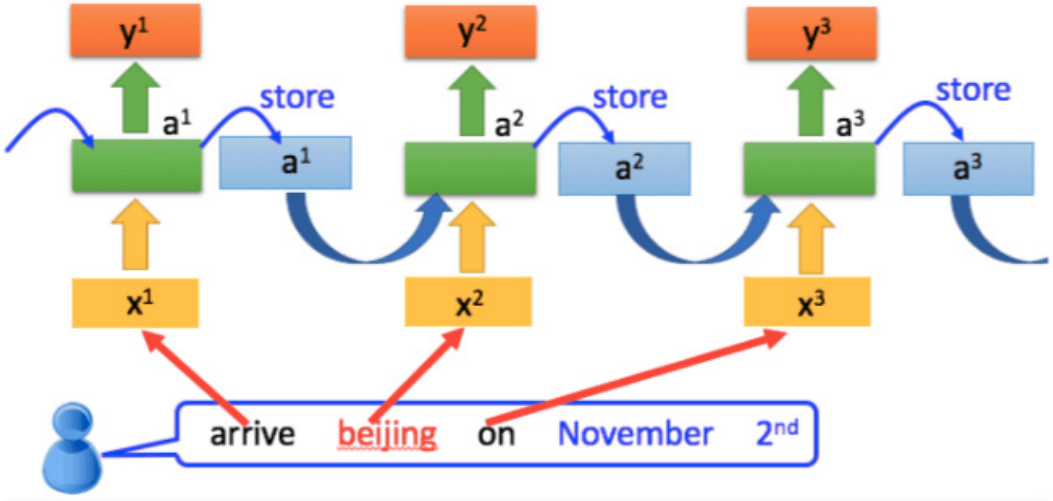
具有记忆效果的网络-循环神经网络

循环神经网络就是这样一种具有记忆效果的网络，下面我们用一个简单的两层网络来举例子



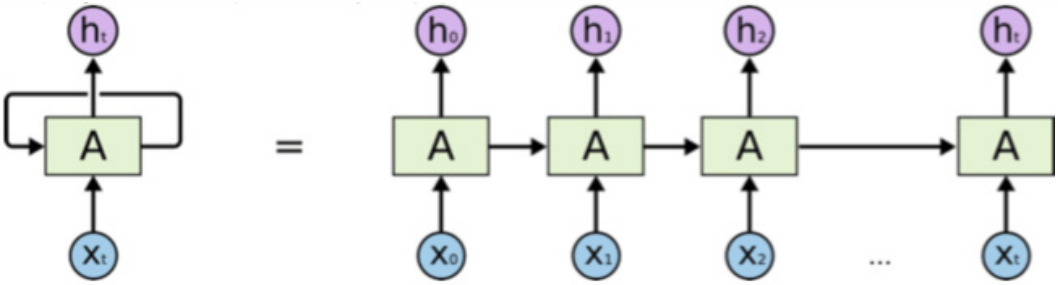
如果除去最左边的部分，那么这就是一个简单的两层神经网络，网络的输入是 x_1 和 x_2，输出是 y_1 和 y_2，如果我们在网络的前向传播中再引入额外的信息，比如 a_1 和 a_2，那么网络就会得到额外的特征输入，从而就算输入是一样的，只要额外的特征输入不同，也会导致最后的结果产生不同的结果，这也就是我们所说的记忆特性。

一般我们讲记忆性，都是对一连串发生的事情来讲，所以循环神经网络一般是使用在一个序列上的，上面这是序列中的一个数据点的过程，下面对于整个序列，有如下的示意图



非常简单，不断将序列中的数据点传入网络，不断这个序列多长，不断循环下去就可以了。这里可能你会有一个疑问，如果序列很长，不断循环下去是不是参数太多了，而且序列的长度如果不定，那么模型的参数一会儿多，一会儿少，这听上去也挺不靠谱的。

事实上，这里用了参数共享，也就是说上面虽然有 3 个绿色的格子，但是实际上它们都是同一个绿色的格子，我们可以用下面的图来形象的表示

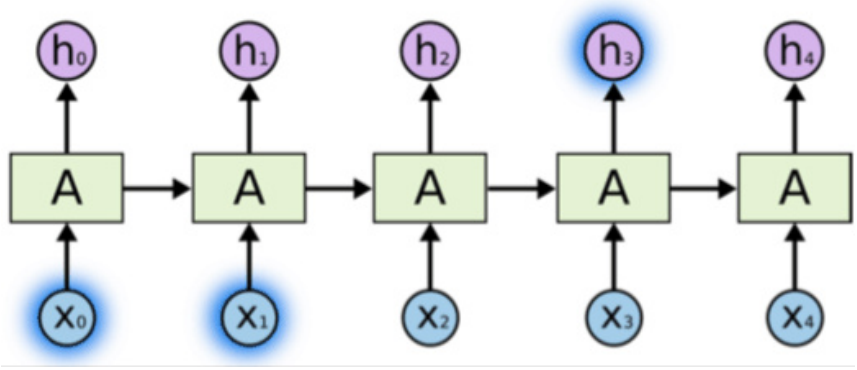


左边就是循环神经网络实际的网络流，右边是将其展开的结果，可以看到网络中的具有循环结构，这也是循环神经网络名字的由来。同时根据循环神经网络的结构我们也可以看出其对于序列类型的数据具有天然的优势，因为网络本身就是一个序列结构，这也是所有循环神经网络最本质的结构。

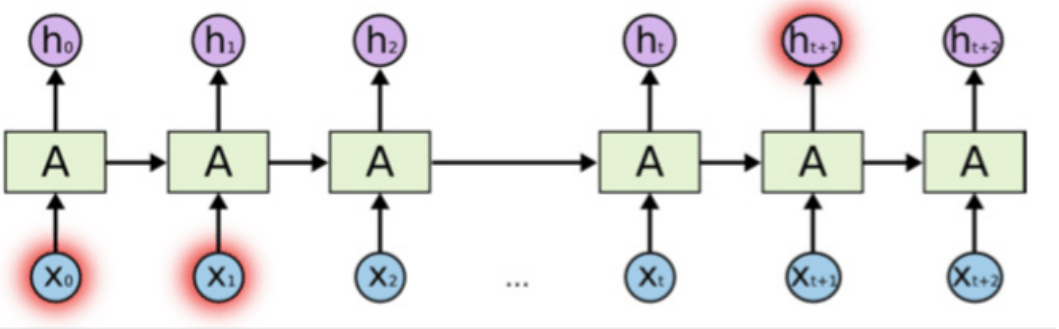
存在的问题

这里提出的这种结构具有记忆的特点，但是随后人们发现网络的记忆能力并没有我们想象中的那么有效，因为伴随着记忆，就有遗忘出现，比如我们总会更加清楚地记得最近发生的事情而遗忘很久之前的事情。

下面我们举个例子，'我住在中国，我会讲**中文**'，那么使用循环神经网络能够非常好的利用'中国'来预测'中文'，就像下面这样



但是如果对于长时依赖的问题，循环神经网络就没有那么有效了，比如我们将上面这句话拆开，第一句话放到文章开头，第二句放到文章的结尾，那么因为记忆的信息和预测位置之间跨度太大，网络就没有办法那么好的记忆长时间的信息了，就像下面这样



正因为存在着这些问题，导致循环神经网络在实际应用上一直没有办法取得很好的效果，但是这个问题能不能解决呢?理论上讲循环神经网络是完全能够解决这种长时依赖问题的，可以针对具体的问题，人为挑选一些特定的参数来解决此问题，但是这并不具有普适性，因为循环神经网络 没有办法自己决定如何该挑选哪些参数。在 1994 年 Bengio 甚至发表论文论述长时依赖问题的原因以及为什么其难以解决。

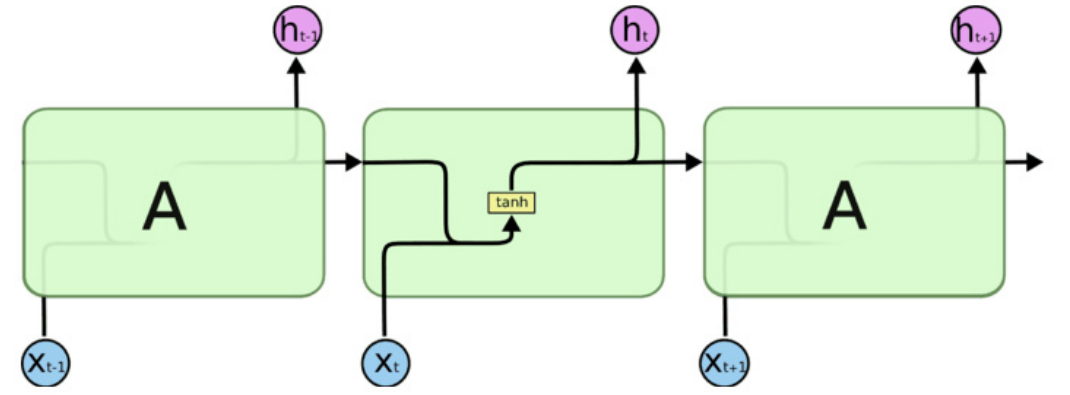
早期循环神经网络的发展存在这长时依赖问题，后来依据循环神经网络的基本结构，有人提出了一些特定的变式，这些变式解决了长时依赖的问题，并在之后循环神经网络的使用中成为主流，下面我们就来讲讲目前循环神经网络最为流行的两种结构，LSTM 和 GRU。

LSTM 和 GRU

前面我们讲到循环神经网络存在的长时依赖问题，后面出现了两种循环神经网络的变式 LSTM 和 GRU，这两种模型随后变得非常流行，基本现在指的循环神经网络都是说的这两种网络，首先我们介绍一个最基本的 RNN。

标准 RNN

标准的 RNN 非常简单，我们可以看看下面的图示



对于每个输入和隐藏状态，标准的 RNN 使用下面的公式进行计算

$$h_t = tanh(w_{ih}x_t + b_{ih} + w_{hh}h_{t-1} + b_{hn}) \tag{1}$$

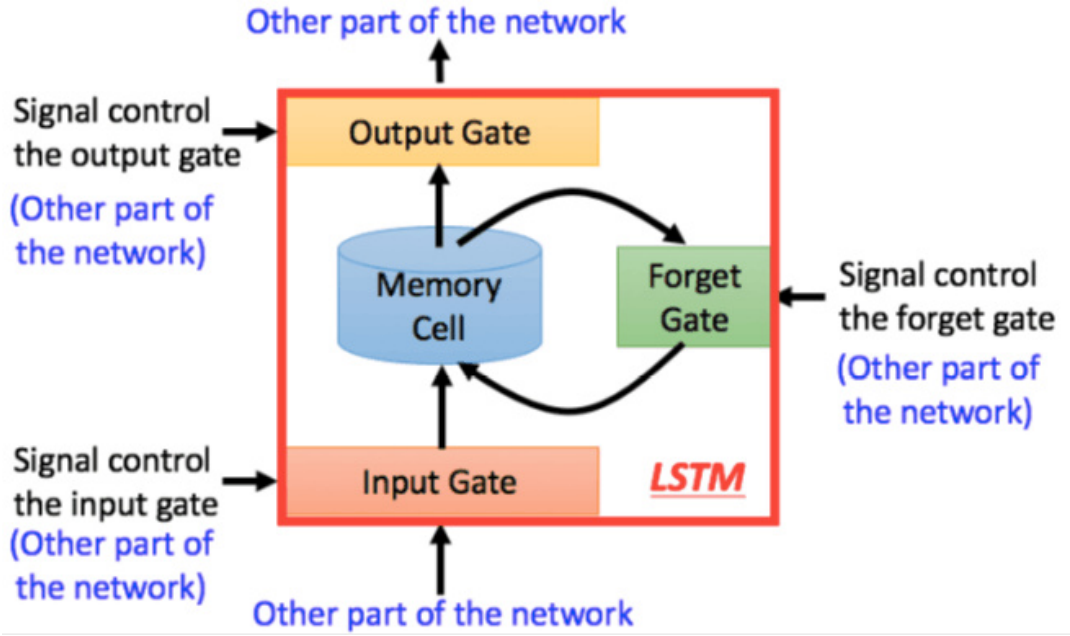
其中 h_{t-1} 表示前一步的记忆状态，x_t 表示当前步的输入，最后得到输出 h_t，同时将其传入的后面作为这一步的记忆状态。

这个标准 RNN 非常简单，所以存在着很多问题，下面我们看看 LSTM。

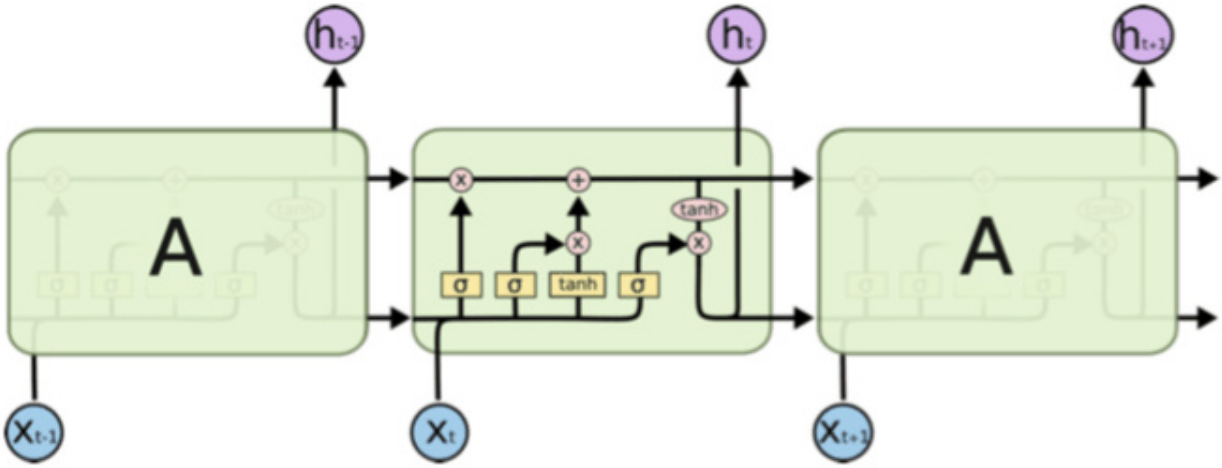
LSTM

LSTM 是 Long Short Term Memory networks 的缩写，能够在一定程度上解决长时依赖的问题。

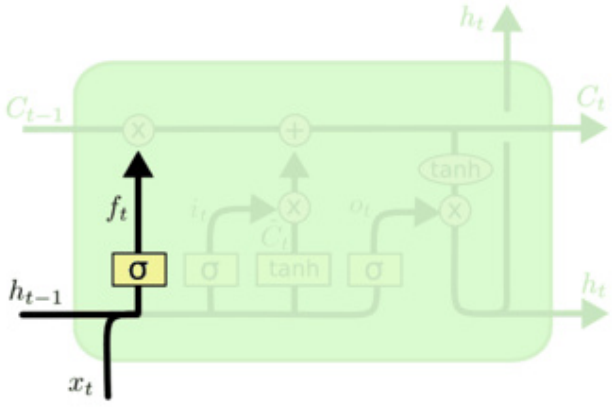
网络的结构和之前讲的 RNN 是一样的，但是内部有着更加复杂的结构，对于单步的抽象网络，我们有下面的图示



LSTM 由三个门来控制，分别是输入门，遗忘门和输出门。输入门和输出门限制着输入和输出的大小，而遗忘门控制着记忆的保留读，对于一个任务，遗忘门能够自己学习到该保留多少以前的记忆，不需要人为进行干扰，所以具备着长时记忆的功能，下面我们看看内部结构

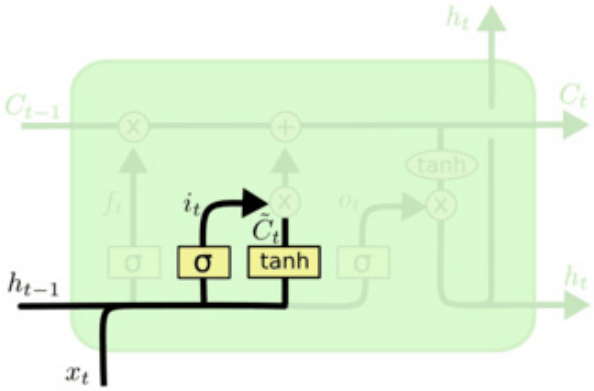


这里是三个序列的 LSTM 结构，每个 A 中的参数都是相同的，下面我们来一步一步讲讲。

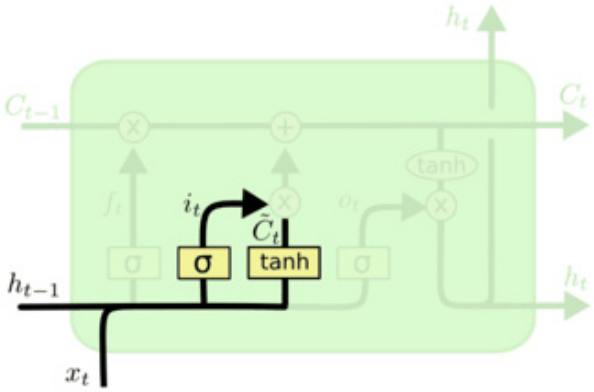


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

首先我们结合 t-1 时刻的网络输出和 t 时刻的网络输入，通过线性变换和 sigmoid 函数得到一个 0 ~ 1 之间的值 f_t，这个值可以称为衰减系数，表示保留过去信息的多少。

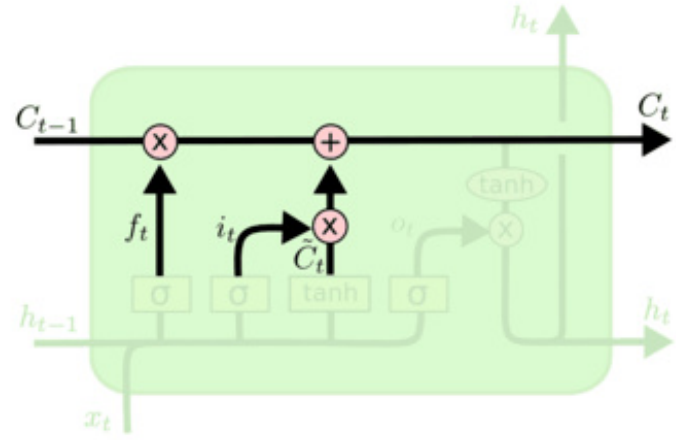


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$



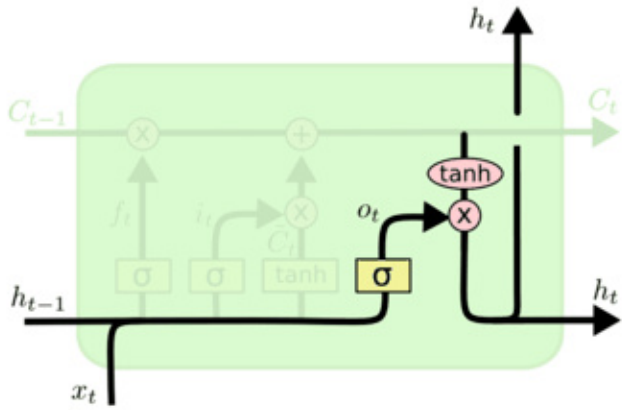
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

然后还是结合 t-1 时刻的输出和 t 时刻的输入分别计算出另外一个系数 i_t 和新接受的信息 $\tilde{C}_t$，其中 i_t 表示保留接受的新信息的多少。



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

然后将前面得到的两个衰减系数分别乘上过去的信息和现在的新信息来确定 t 时刻真正的记忆状态 C_t。



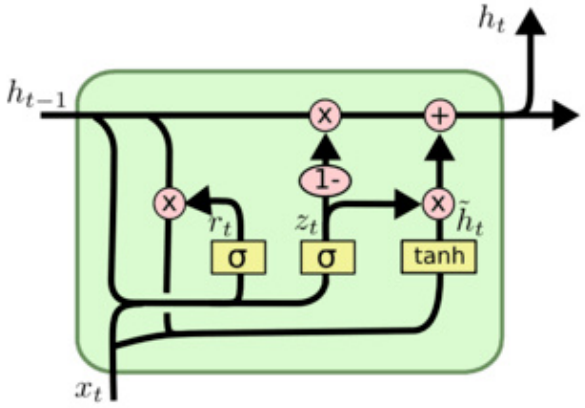
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

最后还是使用 t-1 时刻的输出和 t 时刻的输入计算一个系数，这个系数表示 t 时刻到底输出多少的记忆状态 C_t 得到真正的模型输出 h_t

这就是 LSTM 的整个流程，非常的清楚，也详细说明了三个系数是如何控制输入门，输出门和记忆们。

GRU

下面我们该说说 GRU 了，GRU 是 Gated Recurrent Unit 的缩写，由 2014 年 Cho 提出，GRU 和 LSTM 最大的不同在于 GRU 将遗忘门和术入门合成了一个更新门，同时网络不再额外给出记忆状态 C_t，而是将输出结果作为记忆状态不断往后传，下面是其结构图



$$z_t = \sigma\left(W_z \cdot \left[h_{t-1}, x_t\right]\right)$$
$$r_t = \sigma\left(W_r \cdot \left[h_{t-1}, x_t\right]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot \left[r_t * h_{t-1}, x_t\right]\right)$$
$$h_t = \left(1 - z_t\right) * h_{t-1} + z_t * \tilde{h}_t$$

这里我们就不再分别细说每一步，本质上跟前面的 LSTM 是相同的