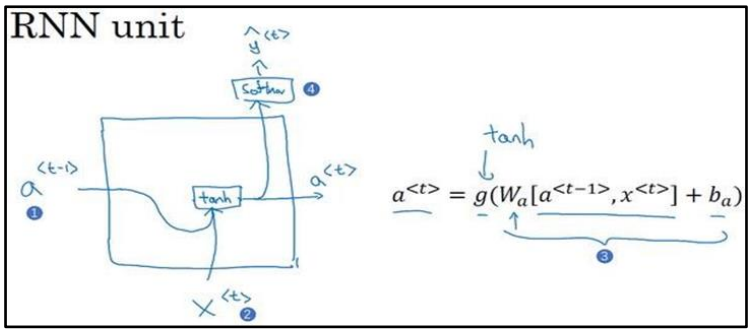
说明：

以下内容均参考自吴恩达《深度学习》课程中的“循环序列模型”一课，而本文就GRU和$LSTM$展开具体地介绍和说明。

Gated Recurrent Unit（GRU）

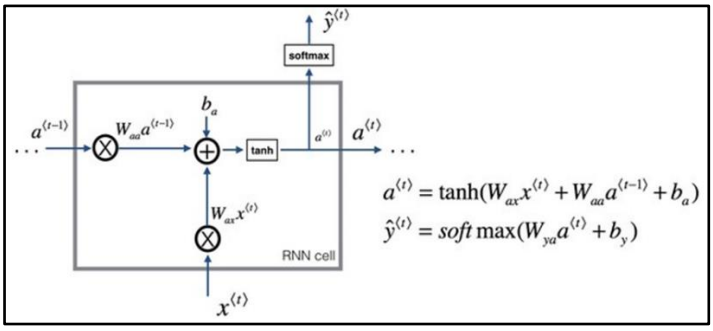
在我们已经了解了基础的RNN模型的运行机制后，下面我们就将来学习**门控循环单元 GRU**，它改变了RNN的隐藏层，使其可以更好地捕捉深层连接，同时也有效改善了梯度消失的问题。



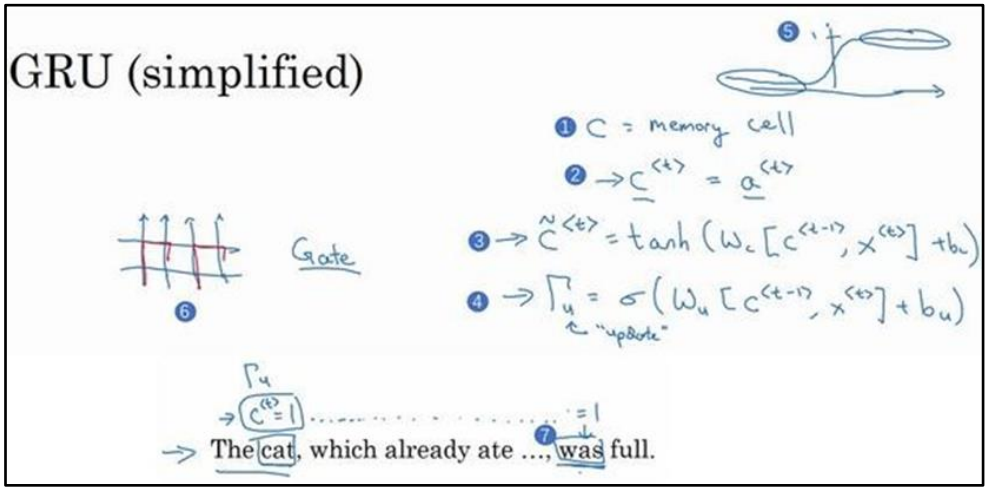
如上图所示，在之前，我们已经学习过了这个公式：

$$a^{<t>} = g_1(W_a[a^{<t-1>}, x^{<t>}] + b_a)$$

在RNN的时间t处，计算其激活值$a^{<t>}$。输入$a^{<t-1>}$（如上图编号 1 所示），即上一个时间步的激活值，再输入$x^{<t>}$（如上图编号 2 所示），然后把这两个并起来，乘上权重项（如上图编号 3 所示）。而如果我们假设g_1是一个$tanh$激活函数，那么在经过$tanh$计算之后，它就会计算出激活值$a^{<t>}$。然后，激活值$a^{<t>}$将会传到$softmax$单元（如上图编号 4 所示）或者其他用于产生输出$\hat{y}^{<t>}$的东西。



就这张图而言，这就是RNN隐藏层的单元的可视化呈现。而之所以展示这张图，是因为下面我们将使用相似的图来讲解门控循环单元GRU。我们还是以“The cat, which already ate……, was full.”句子为例，其中猫是单数的。而为了确保我们已经理解了为什么这里是 was 而不是 were。我们需要知道基本语法：“The cat was full.”或者是“The cats were full”。因此，当我们从左到右读这个句子时，GRU单元将会有个新的变量称为c，代表**细胞（cell）**，即**记忆细胞**（如下图编号 1 所示）。



而记忆细胞的作用是提供了记忆的能力，比如说一只猫是单数还是复数，所以当它看到之后句子的时候，它仍能够判断句子的主语是单数还是复数。于是，在时间t处，有记忆细胞$c^{<t>}$。然后我们可以看到，GRU实际上输出了激活值$a^{<t>}$，$c^{<t>} = a^{<t>}$（如上图编号 2 所示）。于是，我们想要使用不同的符号c和a来表示记忆细胞的值和输出的激活值，即使它们是一样的。（**说明：**我们现在使用上述标记是因为当我们之后说到$LSTMs$的时候，这两个会是不同的值，但是现在对于GRU而言，$c^{<t>}$的值等于$a^{<t>}$的激活值。

因此，下述这些等式就表示了GRU单元的计算。在每个时间步，我们将用一个候选值重写记忆细胞，即$\tilde{c}^{<t>}$的值，所以它就是个候选值，替代了$c^{<t>}$的值。然后，我们用$tanh$激活函数来进行计算，$\tilde{c}^{<t>} = tanh(W_c[c^{<t-1>}, x^{<t>}] +$

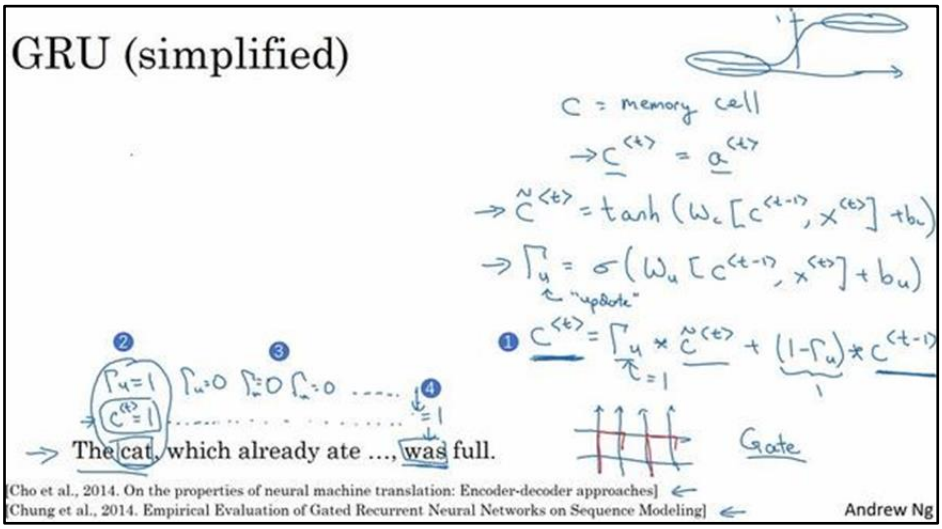
b_c)，所以$\tilde{c}^{<t>}$的值就是个替代值，代替表示$c^{<t>}$的值（如上图编号 3 所示）。

而下面才是重点：在GRU中真正重要的思想是我们有一个叫做Γ_u的门（如上图编号 4 所示），它是个下标为u的大写希腊字母Γ，其中u代表更新门。而Γ_u是一个 0 到 1 之间的值，其计算公式为：

$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$

其中，σ代表的是$sigmoid$函数，而我们还记得$sigmoid$函数是如上图编号 5 所示这样的，它的输出值总是在 0 到 1 之间，且对于大多数可能的输入，$sigmoid$函数的输出则总是非常接近 0 或者非常接近 1。因此，在这样的直觉下，我们可以想到Γ_u在大多数的情况下也是非常接近 0 或 1 的。**除此之外，这个下标字母u表示的是“update”。**然后，GRU的关键部分就是上图编号 3 所示的等式，即我们刚才写出来的用$\tilde{c}$ 更新c 的等式。然后，我们用Γ_u门决定是否要真的更新它。

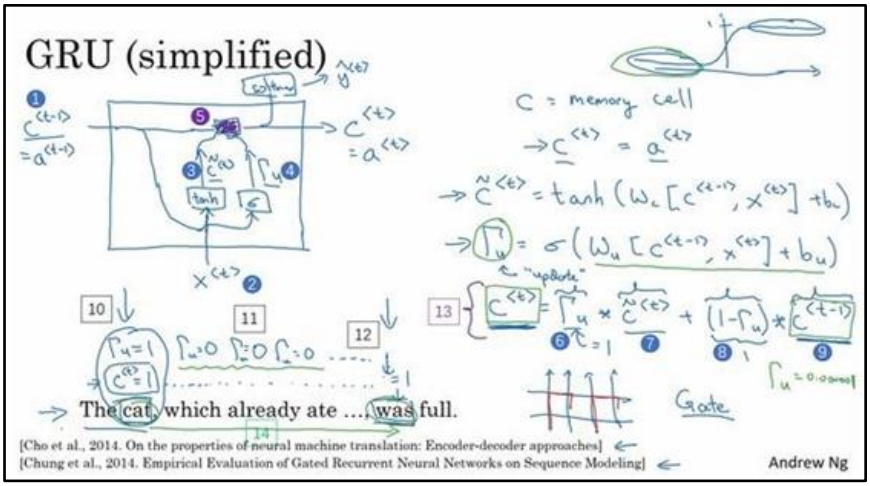
我们也可以这么看待Γ_u，记忆细胞$c^{<t>}$将被设定为 0 或者 1，这取决于我们考虑的单词在句子中是单数还是复数，而因为这里是单数的情况，所以我们先假定它被设为了 1，或者如果是复数的情况我们就把它设为 0。然后，GRU单元将会一直记住$c^{<t>}$的值，直到上图编号 7 所示的位置，$c^{<t>}$的值还是 1，而这就告诉它，噢，这是单数，所以我们用 was。于是门Γ_u的作用就是决定什么时候我们会更新$c^{<t>}$，特别是当我们看到词组 the cat，即句子的主语猫，这就是一个好时机去更新这个值。然后，当我们在使用完它的时候，“The cat, which already ate……, was full.”，我们就知道已经不需要记住它了，就可以忘记它了。



所以，我们接下来要给GRU用的式子就是：

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

如上图编号 1 所示，而如果这个更新值$\Gamma_u = 1$，也就是说把这个新值，即$c^{<t>}$，设为候选值（**说明**：$\Gamma_u = 1$时可简化上式，$c^{<t>}$=$\tilde{c}^{<t>}$）。于是，我们将这个门值设为 1（如上图编号 2 所示），然后往前再更新这个值。而对于所有在这中间的值，我们应该把其门值都设为 0，即$\Gamma_u = 0$，意思就是说不更新它，就采用旧的值。因为如果$\Gamma_u = 0$，则$c^{<t>} = c^{<t-1>}$，其中$c^{<t-1>}$就代表的是旧的值。甚至我们从左到右扫描这个句子，当门值为 0 的时候（如上图编号 3 所示，中间$\Gamma_u = 0$一直为 0，表示一直不更新），也就是说不更新它的时候，我们就用旧的值。这样，即使我们一直处理句子直到上图编号 4 所示的位置，$c^{<t>}$应该也会一直等于$c^{<t-1>}$，于是它会仍然记得猫是单数的。



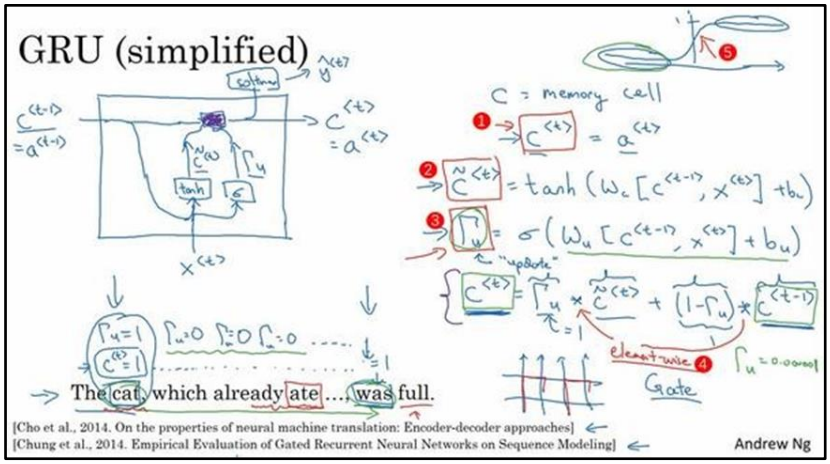
总结一下，GRU单元首先输入$c^{<t-1>}$（如上图编号 1 所示），我们先假设它刚好等于$a^{<t-1>}$，所以我们把这个作为输入。然后$x^{<t>}$也作为输入（如上图编号 2 所示），然后我们把这两个输入用合适的权重结合在一起，再用$tanh$激活函数计算，算出$\tilde{c}^{<t>}$，$\tilde{c}^{<t>} = tanh(W_c[c^{<t-1>}, x^{<t>}] + b_c)$，即$c^{<t>}$的替代值。

然后，我们再用一个不同的参数集，通过$sigmoid$激活函数算出Γ_u，$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$，即更新门。最后，所有的值再通过另一个运算符结合（如上图编号 5 所示），而其所代表的运算过程即上图编号 13 所示的等式。所以，这就是紫色运算符所表示的式子，它输入一个门值Γ_u（如上图编号 6 所示），一个新的候选值$\tilde{c}^{<t>}$（上图编号 7 所示），然后再有一个门值Γ_u（如上图编号 8 所示）和$c^{<t>}$的旧值$c^{<t-1>}$（如上图编号 9 所示）。最后，一起产生记忆细胞的新值$c^{<t>}$，所以，$c^{<t>}$等于$a^{<t>}$。而如果我们想，我们也可以也把它带入$softmax$或者其他预测$y^{<t>}$的东西。

以上就是GRU单元或者说是一个简化过的GRU单元。它的优点就是通过门决定，当我们从左到右扫描一个句子的时候（如上图编号 10 所示），这个时机是要更新某个记忆细胞还是不更新（如上图编号 11 所示，中间Γ_u一直为 0，则表示一直不更新），直到我们真的需要使用记忆细胞的时候（如上图编号 12 所示），而这可能在之前的句子就早已决定了。所以，在这样的情况下，上述的更

新式子（上图编号 13 所示的等式）就会变成$c^{<t>} = c^{<t-1>}$，这非常有利于维持细胞的值。因为Γ_u很接近0，可能是0.000001或者更小，这就不会有梯度消失的问题了。而正是因为Γ_u很接近 0，这就是说$c^{<t>}$几乎就等于$c^{<t-1>}$，而且$c^{<t>}$的值也很好地被维持了，即使经过很多很多的时间步（如上图编号 14 所示）。

总之，这就是缓解梯度消失问题的关键，因此 GRU 允许神经网络运行在非常庞大的依赖词上，比如说 cat 和 was 单词即使被中间的很多单词分割开。



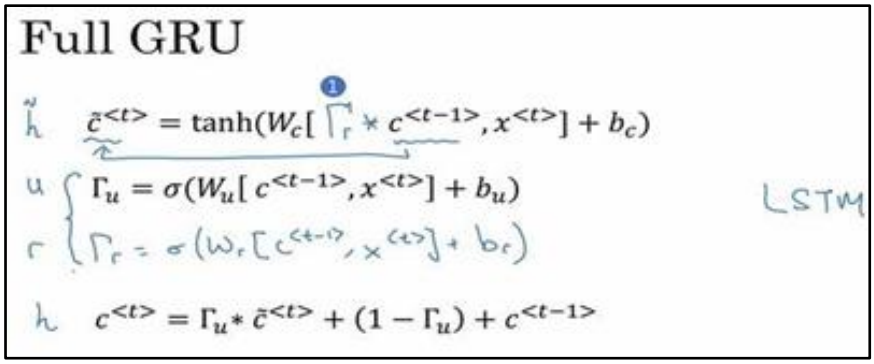
现在，我们再来说下一些实现的细节。在式子 1 中，$c^{<t>}$可以是一个向量（如上图编号 1 所示），如果我们有 100 维的隐藏激活值的话，那么$c^{<t>}$也是 100 维的，$\tilde{c}^{<t>}$也是相同的维度（$\tilde{c}^{<t>} = tanh(W_c[c^{<t-1>}, x^{<t>}] + b_c)$），$\Gamma_u$也是相同的维度（$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$），还有画在框中的其他值。此外，需要注意的是，在这式子中（如上图编号 4 所示），其中的“*”实际上代表的是元素对应的乘积（$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$）。所以，如果说式子中的门$\Gamma_u$是一个 100 维的向量（其里面的值几乎都是 0 或者 1），那么就是说这 100 维的记忆细胞$c^{<t>}$（$a^{<t>} = c^{<t>}$，如上图编号 1 所示）就是我们要更新的比特。

当然，在实际应用中Γ_u不会真的等于 0 或者 1，有时候它会是 0 到 1 的一个中间值（如上图编号 5 所示）。**此外，元素对应的乘积做的就是告诉*GRU*单**

元哪个记忆细胞的向量维度在每个时间步要做更新。所以，我们可以选择保存一些比特不变，而去更新其他的比特。比如说我们可能需要一个比特来记忆猫是单数还是复数的，其他比特来理解我们正在谈论食物还是吃饭，于是，我们可以在每个时间步只改变某一些比特的值。

到现在，我们已经理解了GRU中最重要的思想了，但是，上述过程其实只是简化过的GRU单元，所以现在我们再来描述一下完整的GRU单元。而对于完整的GRU单元，我们要做的一个改变就是在我们计算的第一个式子中给记忆细胞的新候选值$\tilde{c}^{<t>}$加上一个新的项，即添加一个门Γ_r（如下图编号 1 所示），我们也可以认为其中的下标r代表相关性（relevance）。而这个Γ_r门就是告诉我们计算出的下一个$c^{<t>}$的候选值$\tilde{c}^{<t>}$跟$c^{<t-1>}$有多大的相关性。此外，计算门Γ_r也需要参数，因此，它的计算公式为：

$$\Gamma_r = \sigma(W_r[c^{<t-1>}, x^{<t>}] + b_r)$$



总之，正如我们所见，其实有很多方法可以来设计这些类型的神经网络。而多年来研究者们也试验过很多很多不同可能的方法来设计这些单元，去尝试让神经网络有更深层的连接，去尝试产生更大范围的影响，还有解决梯度消失的问题，但GRU就是其中一个研究者们最常使用的版本，它也被发现在很多不同的问题上也是非常健壮和实用的。当然，我们也可以尝试发明新类型的单元，只要我们愿意。**但是，GRU是一个标准版本，也就是最常使用的。**然后，另一个常用的版本被称为***LSTM***，即长短时记忆网络，而我们就将在下节内容中对它展开介绍。

LSTM（long short term memory）unit

在上节中我们已经学习了GRU（门控循环单元），它能够让我们在序列中学习非常深的连接。但其他类型的单元也可以让我们做到这个，比如说 LSTM，即长短时记忆网络，它甚至比GRU更加有效。

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_r = \sigma(W_r[c^{<t-1>}, x^{<t>}] + b_r)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

$$a^{<t>} = c^{<t>}$$

如上图所示，对于GRU我们有$a^{<t>} = c^{<t>}$。 此外，还有两个门：

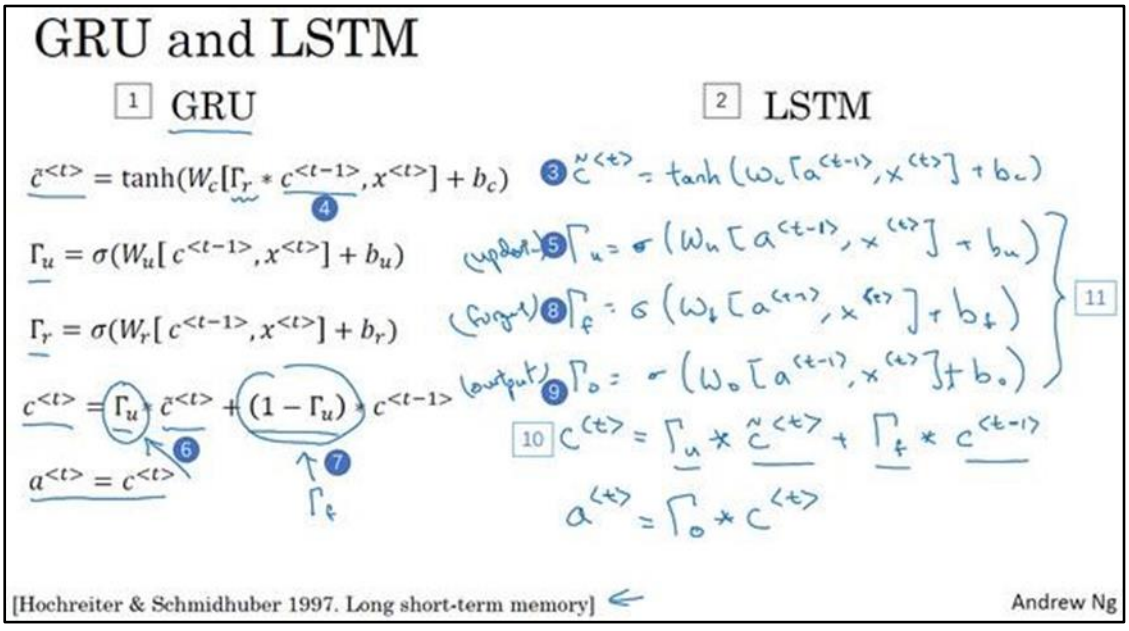
更新门Γ_u（the update gate）：$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$

相关门Γ_r（the relevance gate）：$\Gamma_r = \sigma(W_r[c^{<t-1>}, x^{<t>}] + b_r)$

$\tilde{c}^{<t>} = tanh(W_c[c^{<t-1>}, x^{<t>}] + b_c)$，这是代替记忆细胞的候选值，然后我们使用更新门$\Gamma_u$来决定是否要用$\tilde{c}^{<t>}$来更新$c^{<t>}$：

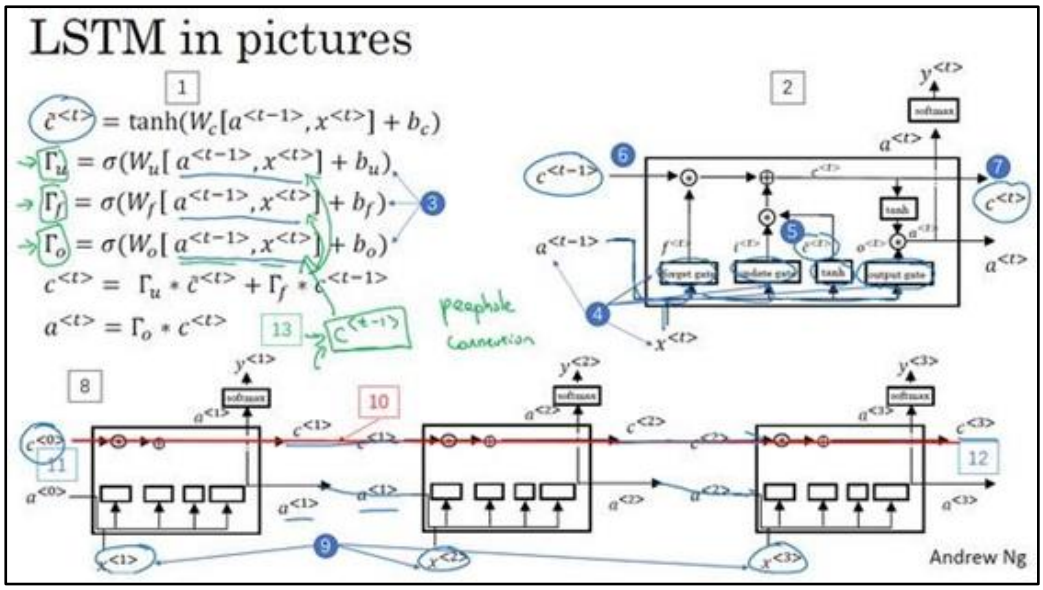
$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

而$LSTM$是一个比GRU更加强大和通用的版本，如下图所示：



这就是$LSTM$主要的式子（上图编号 2 所示）。我们继续回到记忆细胞c上面来，使用$\tilde{c}^{<t>} = tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)$来更新它的候选值$\tilde{c}^{<t>}$（如上图编号 3 所示）。但是要注意，在$LSTM$中我们不再有$a^{<t>} = c^{<t>}$的情况了，这是现在我们用的是类似于左边这个式子（如上图编号 4 所示），但是有一些改变，现在我们专门使用$a^{<t>}$或者$a^{<t-1>}$，而不是用$c^{<t-1>}$，并且我们也不再使用相关门Γ_r，但我们仍像以前那样有一个更新门Γ_u和表示更新的参数W_u，$\Gamma_u = \sigma(W_u[a^{<t-1>}, x^{<t>}] + b_u)$（如上图编号 5 所示）。而一个$LSTM$的新特性是不只有一个更新门控制的，这里的这两项（如上图编号 6、7 所示），我们将用不同的项来代替它们，要用别的项来取代Γ_u和$1 - \Gamma_u$，对于这里（上图编号 6 所示）我们用Γ_u。然后这里（上图编号 7 所示）我们用遗忘门Γ_f（the forget gate），所以这个$\Gamma_f = \sigma\big(W_f[a^{<t-1>}, x^{<t>}] + b_f\big)$（如上图编号 8 所示）。然后，我们还有一个新的输出门Γ_o，$\Gamma_o = \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o)$（如上图编号 9 所示）。于是，记忆细胞的更新值$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$（如上图编号 10 所示）。所以，这给了记忆细胞选择权去维持旧的值$c^{<t-1>}$或者就加上新的值$\tilde{c}^{<t>}$，但这里用了单独的更新门Γ_u和遗忘门Γ_f。而最后，$a^{<t>} = c^{<t>}$的式子会变成$a^{<t>} = \Gamma_o * c^{<t>}$。以上，这就是$LSTM$主要的式子了。

总之，在$LSTM$中，这里（上图编号 11 所示）有三个门而不是两个，所以它有点复杂，因为它把一些门放到了和之前有点不同的地方。

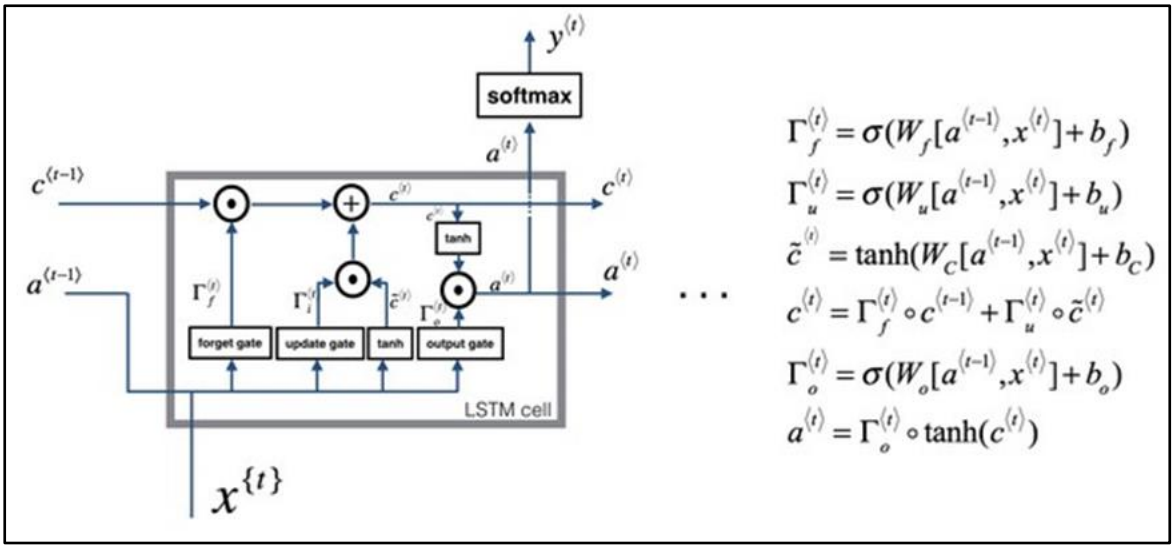


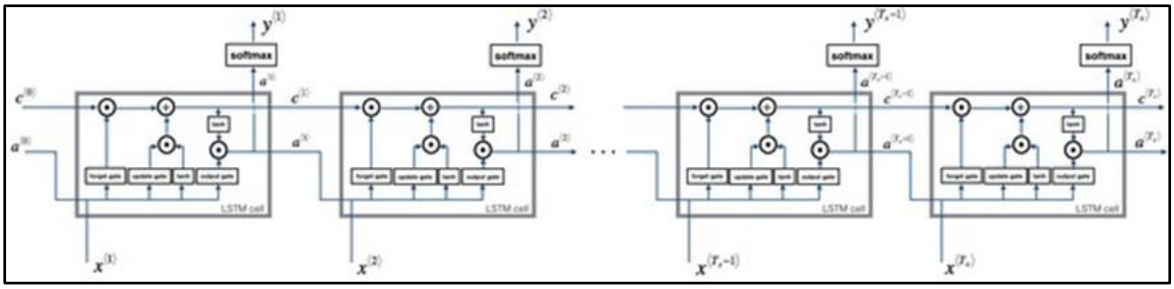
再提一下，以上这些式子就是控制$LSTM$行为的主要的式子了（如上图编号 1 所示）。我们像之前一样再用图片稍微解释一下（如上图编号 2 所示）。在这张图里，我们用了$a^{<t-1>}$和$x^{<t>}$来计算所有门值（如上图编号 3，4 所示），而我们计算了遗忘门Γ_f的值，还有更新门Γ_u以及输出门Γ_o的值（如上图编号 4 所示）。然后，我们也经过$tanh$函数来计算$\tilde{c}^{<t>}$（如上图编号 5 所示）。而上述这些值被用复杂的方式组合在一起，比如说元素对应的乘积或者其他的方式来从之前的$c^{<t-1>}$（如上图编号 6 所示）中获得$c^{<t>}$（如上图编号 7 所示）。

注意：这里的其中一个元素很有意思，如我们在这一堆图（如上图编号 8 所示的一系列图片）中看到的，我们把它们按时间次序连起来，这里（如上图编号 9 所示）输入$x^{<1>}$，$x^{<2>}$，$x^{<3>}$，然后我们可以把这些单元依次连接起来，这里输出了上一个时间的a，而a又会作为下一个时间步的输入，c也同理。此外，我们还会注意到上面这里有条线（如上图编号 10 所示的线），这条线显示了只要我们正确地设置了遗忘门和更新门，$LSTM$是相当容易把$c^{<0>}$的值（如上图编号 11 所示）一直往下传递到右边的，比如$c^{<3>} = c^{<0>}$（如上图编号 12 所示），而这就是为什么$LSTM$和GRU都非常擅长于长时间记忆某个值。

总之，这就是*LSTM*，但我们可能会想到本节所讲的内容和一般使用的*LSTM*版本会有些不同，而最常用的版本可能是门值不仅取决于$a^{<t-1>}$和$x^{<t>}$，有时候也可以偷窥一下$c^{<t-1>}$的值（如上图编号 13 所示），而这又叫做**“窥视孔连接”（peephole connection）**。虽然这不是个好听的名字，但是**“偷窥孔连接”**的意思其实就是门值不仅取决于$a^{<t-1>}$和$x^{<t>}$，也取决于上一个记忆细胞的$c^{<t-1>}$值，然后，“偷窥孔连接”就可以结合这三个门（Γ_u、Γ_f 、Γ_o）来计算了。

***LSTM*的前向传播：**





$$\Gamma_f{}^{<t>} = \sigma\left(W_{fa}a^{<t-1>} + W_{fx}x^{<t>} + b_f\right)$$

$$\Gamma_u{}^{<t>} = \sigma(W_{ua}a^{<t-1>} + W_{ux}x^{<t>} + b_u)$$

$$\tilde{c}^{<t>} = tanh(W_{ca}a^{<t-1>} + W_{cx}x^{<t>} + b_c)$$

$$c^{<t>} = \Gamma_u{}^{<t>} * \tilde{c}^{<t>} + \Gamma_f{}^{<t>} * c^{<t-1>}$$

$$\Gamma_o{}^{<t>} = \sigma(W_{oa}a^{<t-1>} + W_{ox}x^{<t>} + b_o)$$

$$a^{<t>} = \Gamma_o{}^{<t>} * \tanh(c^{<t>})$$

$$\hat{y}^{<t>} = \sigma(W_y a^{<t>} + b_y)$$

LSTM的反向传播计算：

$$\frac{\partial \tanh(x)}{\partial x} = 1 - \tanh(x)^2$$

$$\frac{\partial \sigma(x)}{\partial x} = x(1-x)$$

$$J(\hat{y}, y) = \sum_{t=1}^{T_x} L^{<t>}(\hat{y}^{<t>}, y^{<t>})$$

$$L^{<t>}(\hat{y}^{<t>}, y^{<t>}) = -y^{<t>} log\hat{y}^{<t>} - (1-y^{<t>})\log(1-\hat{y}^{<t>})$$

$$\frac{\partial J}{\partial a^{<t>}} = \frac{\partial J}{\partial L^{<t>}} \cdot \frac{\partial L^{<t>}}{\partial \hat{y}^{<t>}} \cdot \frac{\partial \hat{y}^{<t>}}{\partial a^{<t>}} = W_y^T(\hat{y}^{<t>} - y^{<t>})$$

$$\frac{\partial J}{\partial W_y} += (\hat{y}^{<t>} - y^{<t>}) a^{<t>T}$$

$$\frac{\partial J}{\partial b_y} += \sum_{batch} (\hat{y}^{<t>} - y^{<t>})$$

$$\frac{\partial J}{\partial c^{<t>}} = \frac{\partial J}{\partial a^{<t>}} \frac{\partial a^{<t>}}{\partial c^{<t>}} = \frac{\partial J}{\partial a^{<t>}} \cdot \Gamma_o{}^{<t>} * (1 - \tanh(c^{<t>})^2)$$

在反向传播中，每个时间步t接收来自上一个时间步$t+1$的激活值$a^{<t>}$和记忆细胞$c^{<t>}$的导数$\frac{\partial J}{\partial a^{<t>}}$和$\frac{\partial J}{\partial c^{<t>}}$，则：

$$\frac{\partial J}{\partial c^{<t-1>}} += \frac{\partial J}{\partial c^{<t>}} \frac{\partial c^{<t>}}{\partial c^{<t-1>}} = \frac{\partial J}{\partial c^{<t>}} \cdot \Gamma_f{}^{<t>}$$

$$\frac{\partial J}{\partial \Gamma_f{}^{<t>}} = \frac{\partial J}{\partial c^{<t>}} \frac{\partial c^{<t>}}{\partial \Gamma_f{}^{<t>}} = \frac{\partial J}{\partial c^{<t>}} \cdot c^{<t-1>}$$

$$\frac{\partial J}{\partial \Gamma_u{}^{<t>}} = \frac{\partial J}{\partial c^{<t>}} \frac{\partial c^{<t>}}{\partial \Gamma_u{}^{<t>}} = \frac{\partial J}{\partial c^{<t>}} \cdot \tilde{c}^{<t>}$$

$$\frac{\partial J}{\partial \tilde{c}^{<t>}} = \frac{\partial J}{\partial c^{<t>}} \frac{\partial c^{<t>}}{\partial \tilde{c}^{<t>}} = \frac{\partial J}{\partial c^{<t>}} \cdot \Gamma_u{}^{<t>}$$

$$\frac{\partial J}{\partial \Gamma_o{}^{<t>}} = \frac{\partial J}{\partial a^{<t>}} \frac{\partial a^{<t>}}{\partial \Gamma_o{}^{<t>}} = \frac{\partial J}{\partial a^{<t>}} \cdot \tanh(c^{<t>})$$

$$\frac{\partial J}{\partial W_{fa}} += \frac{\partial J}{\partial \Gamma_f{}^{<t>}} \frac{\partial \Gamma_f{}^{<t>}}{\partial W_{fa}} = \frac{\partial J}{\partial \Gamma_f{}^{<t>}} \cdot \Gamma_f{}^{<t>} \cdot (1 - \Gamma_f{}^{<t>}) a^{<t-1>T}$$

$$\frac{\partial J}{\partial W_{fx}} += \frac{\partial J}{\partial \Gamma_f^{<t>}} \frac{\partial \Gamma_f^{<t>}}{\partial W_{fx}} = \frac{\partial J}{\partial \Gamma_f^{<t>}} \cdot \Gamma_f^{<t>} \cdot (1 - \Gamma_f^{<t>}) x^{<t>T}$$

$$\frac{\partial J}{\partial b_f} += \frac{\partial J}{\partial \Gamma_f^{<t>}} \frac{\partial \Gamma_f^{<t>}}{\partial b_f} = \frac{\partial J}{\partial \Gamma_f^{<t>}} \cdot \sum_{batch} (\Gamma_f^{<t>} \cdot (1 - \Gamma_f^{<t>}))$$

$$\frac{\partial J}{\partial W_{ua}} += \frac{\partial J}{\partial \Gamma_u^{<t>}} \frac{\partial \Gamma_u^{<t>}}{\partial W_{ua}} = \frac{\partial J}{\partial \Gamma_u^{<t>}} \cdot \Gamma_u^{<t>} \cdot (1 - \Gamma_u^{<t>}) a^{<t-1>T}$$

$$\frac{\partial J}{\partial W_{ux}} += \frac{\partial J}{\partial \Gamma_u^{<t>}} \frac{\partial \Gamma_u^{<t>}}{\partial W_{ux}} = \frac{\partial J}{\partial \Gamma_u^{<t>}} \cdot \Gamma_u^{<t>} \cdot (1 - \Gamma_u^{<t>}) x^{<t>T}$$

$$\frac{\partial J}{\partial b_u} += \frac{\partial J}{\partial \Gamma_u^{<t>}} \frac{\partial \Gamma_u^{<t>}}{\partial b_u} = \frac{\partial J}{\partial \Gamma_u^{<t>}} \cdot \sum_{batch} (\Gamma_u^{<t>} \cdot (1 - \Gamma_u^{<t>}))$$

$$\frac{\partial J}{\partial W_{oa}} += \frac{\partial J}{\partial \Gamma_o^{<t>}} \frac{\partial \Gamma_u^{<t>}}{\partial W_{oa}} = \frac{\partial J}{\partial \Gamma_o^{<t>}} \cdot \Gamma_o^{<t>} \cdot (1 - \Gamma_o^{<t>}) a^{<t-1>T}$$

$$\frac{\partial J}{\partial W_{ox}} += \frac{\partial J}{\partial \Gamma_o^{<t>}} \frac{\partial \Gamma_o^{<t>}}{\partial W_{ox}} = \frac{\partial J}{\partial \Gamma_o^{<t>}} \cdot \Gamma_o^{<t>} \cdot (1 - \Gamma_o^{<t>}) x^{<t>T}$$

$$\frac{\partial J}{\partial b_o} += \frac{\partial J}{\partial \Gamma_o^{<t>}} \frac{\partial \Gamma_o^{<t>}}{\partial b_o} = \frac{\partial J}{\partial \Gamma_o^{<t>}} \cdot \sum_{batch} (\Gamma_o^{<t>} \cdot (1 - \Gamma_o^{<t>}))$$

$$\frac{\partial J}{\partial W_{ca}} += \frac{\partial J}{\partial \Gamma_c^{<t>}} \frac{\partial \Gamma_c^{<t>}}{\partial W_{ca}} = \frac{\partial J}{\partial \Gamma_c^{<t>}} \cdot \Gamma_c^{<t>} \cdot (1 - \Gamma_c^{<t>}) a^{<t-1>T}$$

$$\frac{\partial J}{\partial W_{cx}} += \frac{\partial J}{\partial \Gamma_c^{<t>}} \frac{\partial \Gamma_c^{<t>}}{\partial W_{cx}} = \frac{\partial J}{\partial \Gamma_c^{<t>}} \cdot \Gamma_c^{<t>} \cdot (1 - \Gamma_c^{<t>}) x^{<t>T}$$

$$\frac{\partial J}{\partial b_c} += \frac{\partial J}{\partial \Gamma_c^{<t>}} \frac{\partial \Gamma_c^{<t>}}{\partial b_c} = \frac{\partial J}{\partial \Gamma_c^{<t>}} \cdot \sum_{batch} (\Gamma_c^{<t>} \cdot (1 - \Gamma_c^{<t>}))$$

$$\frac{\partial J}{\partial a^{<t-1>}} += \frac{\partial J}{\partial \Gamma_f^{<t>}} \frac{\partial \Gamma_f^{<t>}}{\partial a^{<t-1>}} + \frac{\partial J}{\partial \Gamma_u^{<t>}} \frac{\partial \Gamma_u^{<t>}}{\partial a^{<t-1>}} + \frac{\partial J}{\partial \Gamma_o^{<t>}} \frac{\partial \Gamma_o^{<t>}}{\partial a^{<t-1>}} + \frac{\partial J}{\partial \Gamma_c^{<t>}} \frac{\partial \Gamma_c^{<t>}}{\partial a^{<t-1>}}$$

$$\frac{\partial J}{\partial x^{<t>}} = \frac{\partial J}{\partial \Gamma_f^{<t>}} \frac{\partial \Gamma_f^{<t>}}{\partial x^{<t>}} + \frac{\partial J}{\partial \Gamma_u^{<t>}} \frac{\partial \Gamma_u^{<t>}}{\partial x^{<t>}} + \frac{\partial J}{\partial \Gamma_o^{<t>}} \frac{\partial \Gamma_o^{<t>}}{\partial x^{<t>}} + \frac{\partial J}{\partial \Gamma_c^{<t>}} \frac{\partial \Gamma_c^{<t>}}{\partial x^{<t>}}$$