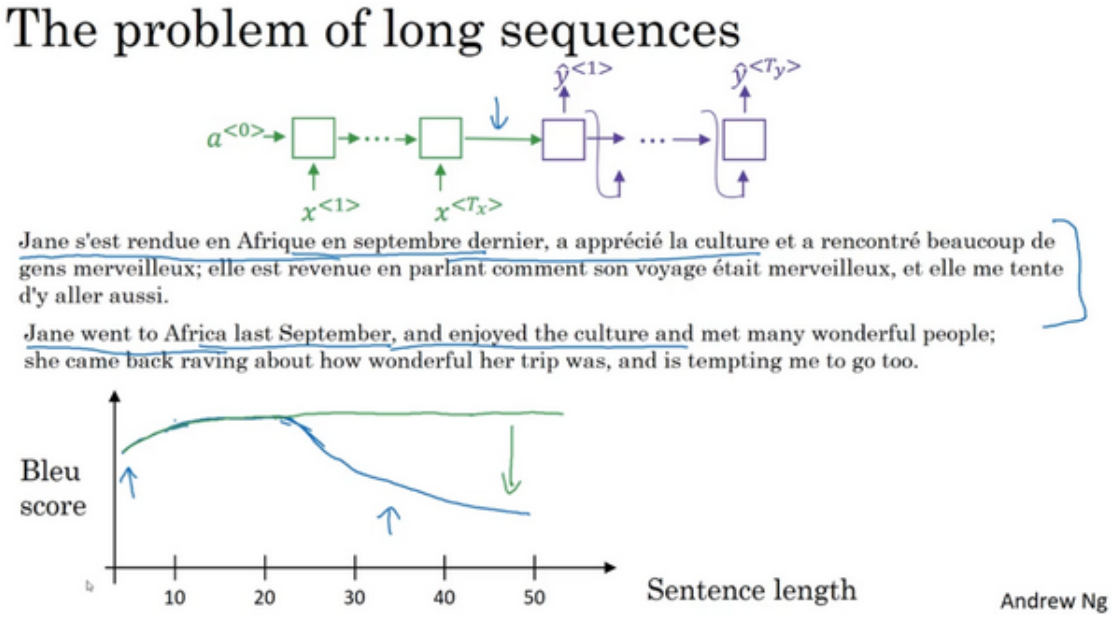
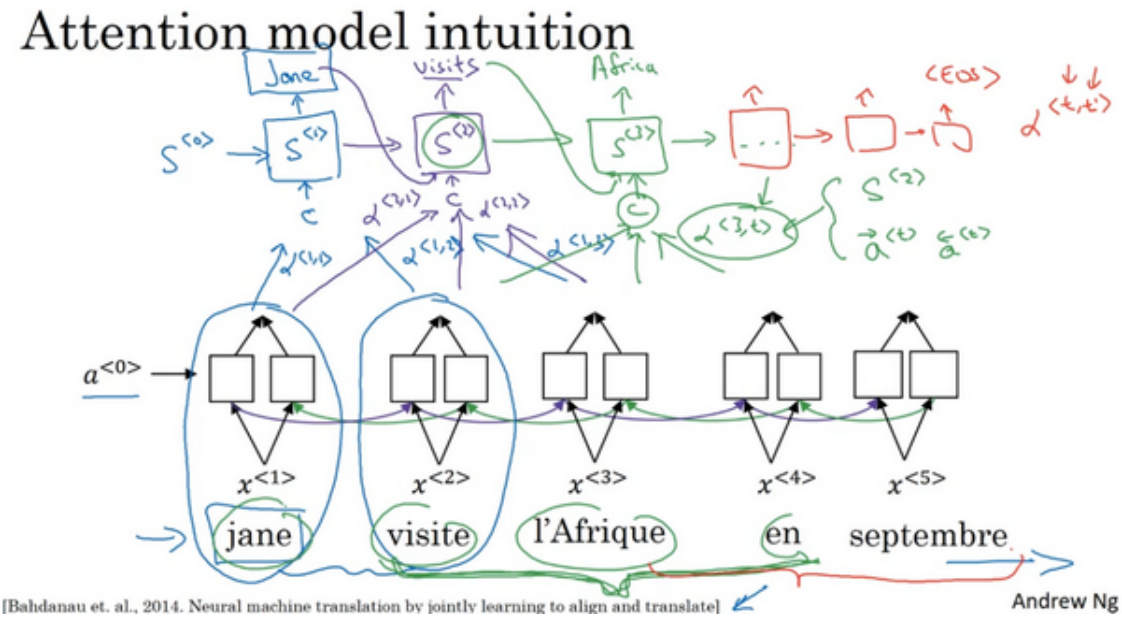
3.7 注意力模型直观理解（Attention Model Intuition）

在本周大部分时间中，你都在使用这个编码解码的构架（**a Encoder-Decoder architecture**）来完成机器翻译。当你使用 **RNN** 读一个句子，于是另一个会输出一个句子。我们要对其做一些改变，称为注意力模型（**the Attention Model**），并且这会使它工作得更好。注意力模型或者说注意力这种思想（**The attention algorithm, the attention idea**）已经是深度学习中最重要的思想之一，我们看看它是怎么运作的。



像这样给定一个很长的法语句子，在你的神经网络中，这个绿色的编码器要做的就是读整个句子，然后记忆整个句子，再在感知机中传递（**to read in the whole sentence and then memorize the whole sentences and store it in the activations conveyed her**）。而对于这个紫色的神经网络，即解码网络（**the decoder network**）将生成英文翻译，**Jane** 去年九月去了非洲，非常享受非洲文化，遇到了很多奇妙的人，她回来就嚷嚷道，她经历了一个多棒的旅行，并邀请我也一起去。人工翻译并不会通过读整个法语句子，再记忆里面的东西，然后从零开始，机械式地翻译成一个英语句子。而人工翻译，首先会做的可能是先翻译出句子的部分，再看下一部分，并翻译这一部分。看一部分，翻译一部分，一直这样下去。你会通过句子，一点一点地翻译，因为记忆整个的像这样的的句子是非常困难的。你在下面这个编码解码结构中，会看到它对于短句子效果非常好，于是它会有一个相对高的 **Bleu** 分（**Bleu score**），但是对于长句子而言，比如说大于 30 或者 40 词的句子，它的表现就会变差。**Bleu** 评分看起来就会像是这样，随着单词数量变化，短的句子会难以翻译，因为很难得到所有词。对于长的句子，

效果也不好，因为在神经网络中，记忆非常长句子是非常困难的。在这个和下个视频中，你会见识到注意力模型，它翻译得很像人类，一次翻译句子的一部分。而且有了注意力模型，机器翻译系统的表现会像这个一样，因为翻译只会翻译句子的一部分，你不会看到这个有一个巨大的下倾（**huge dip**），这个下倾实际上衡量了神经网络记忆一个长句子的能力，这是我们不希望神经网络去做的事情。在这个视频中，我想要给你们注意力机制运行的一些直观的东西。然后在下个视频中，完善细节。



注意力模型源于 **Dimitri, Bahdanau, Camcrun Cho, Yoshe Bengio**。（**Bahdanau D, Cho K, Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate[J]. Computer Science, 2014.**）虽然这个模型源于机器翻译，但它也推广到了其他应用领域。我认为在深度学习领域，这个是个非常有影响力的，非常具有开创性的论文。

让我们用一个短句举例说明一下，即使这些思想可能是应用于更长的句子。但是用短句来举例说明，讲解这些思想会更简单。我们有一个很平常的句子：(法语)**Jane visite l'Afrique en Septembre**。假定我们使用 **RNN**，在这个情况中，我们将使用一个双向的 **RNN**（**a bidirectional RNN**），为了计算每个输入单词的的特征集（**set of features**），你必须要理解输出$\hat{y}^{<1>}$到$\hat{y}^{<3>}$一直到$\hat{y}^{<5>}$的双向 **RNN**。但是我们并不是只翻译一个单词，让我们先去掉上面的Y，就用双向的 **RNN**。我们要对单词做的就是，对于句子里的每五个单词，计算一个句子中单词的特征集，也有可能是周围的词，让我们试试，生成英文翻译。我们将使用另一个 **RNN** 生成英文翻译，这是我平时用的 **RNN** 记号。我不用A来表示感知机（**the activation**），这是为了避免和这里的感知机（**the activations**）混淆。我会用另一个不同的记号，我会用S

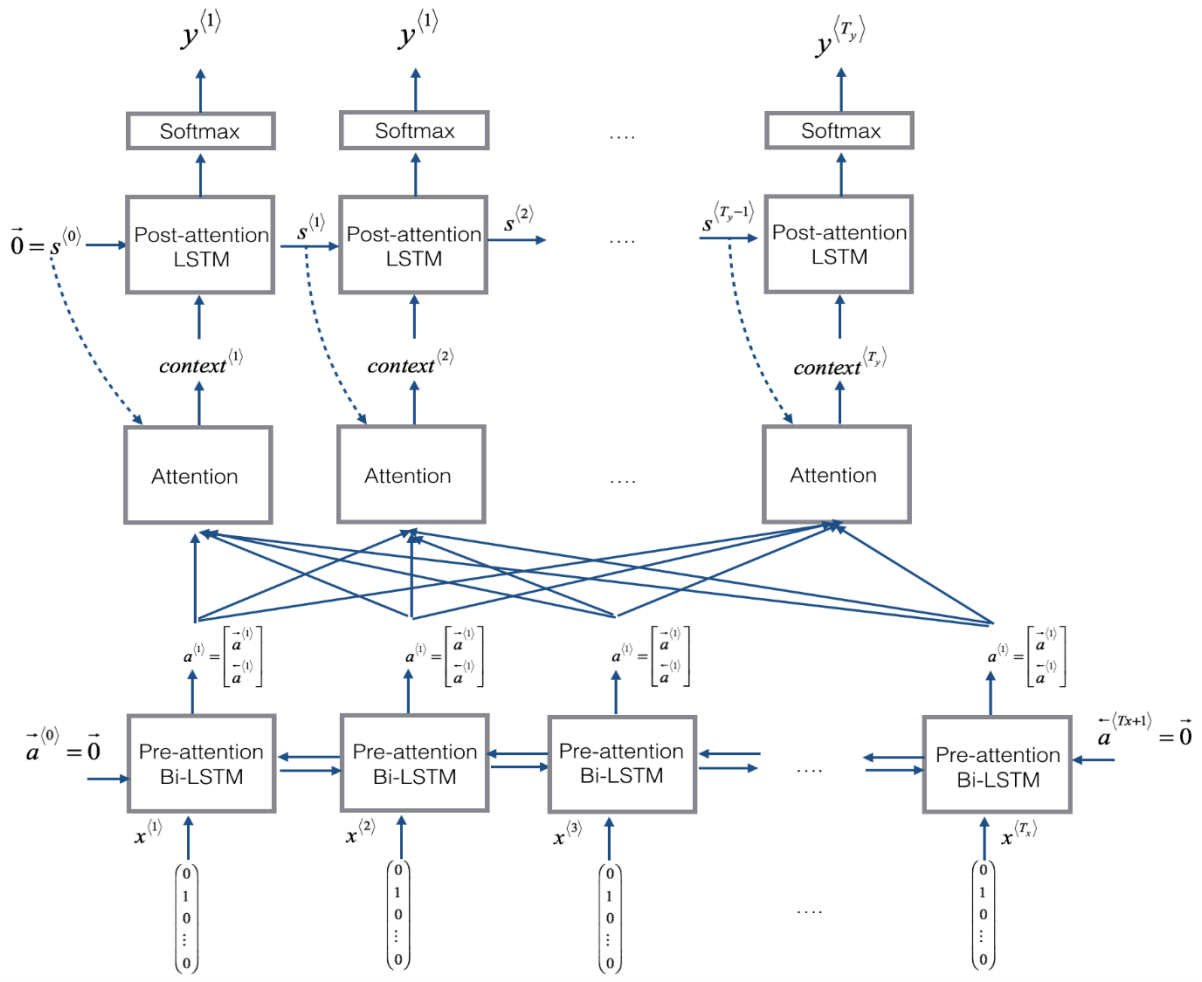
来表示 RNN 的隐藏状态（the hidden state in this RNN），不用$A^{<1>}$，而是用$S^{<1>}$。我们希望在这个模型里第一个生成的单词将会是 **Jane**，为了生成 **Jane visits Africa in September**。于是等式就是，当你尝试生成第一个词，即输出，那么我们应该看输入的法语句子的哪个部分？似乎你应该先看第一个单词，或者它附近的词。但是你别看太远了，比如说看到句尾去了。所以注意力模型就会计算注意力权重（a set of attention weights），我们将用$a^{<1,1>}$来表示当你生成第一个词时你应该放多少注意力在这个第一块信息处。然后我们算第二个，这个叫注意力权重，$a^{<1,2>}$它告诉我们当你尝试去计算第一个词 **Jane** 时，我们应该花多少注意力在输入的第二个词上面。同理这里是$a^{<1,3>}$，接下去也同理。这些将会告诉我们，我们应该花多少注意力在记号为C的内容上。这就是 **RNN** 的一个单元，如何尝试生成第一个词的，这是 **RNN** 的其中一步，我们将会在下个视频中讲解细节。对于 **RNN** 的第二步，我们将有一个新的隐藏状态$S^{<2>}$，我们也会用一个新的注意力权值集(a new set of the attention weights),我们将用$a^{<2,1>}$来告诉我们什么时候生成第二个词，那么 **visits** 就会是第二个标签了(**the ground trip label**)。我们应该花多少注意力在输入的第一个法语词上。然后同理$a^{<2,2>}$，接下去也同理，我们应该花多少注意力在 **visite** 词上，我们应该花多少注意在词 **l'Afique** 上面。当然我们第一个生成的词 **Jane** 也会输入到这里，于是我们就有了需要花注意力的上下文。第二步，这也是个输入，然后会一起生成第二个词，这会让我们来到第三步$S^{<3>}$，这是输入，我们再有上下文 **C**，它取决于在不同的时间集（**time sets**）,上面的$a^{<3>}$。这个告诉了我们我们要花注意力在不同的法语的输入词上面。 然后同理。

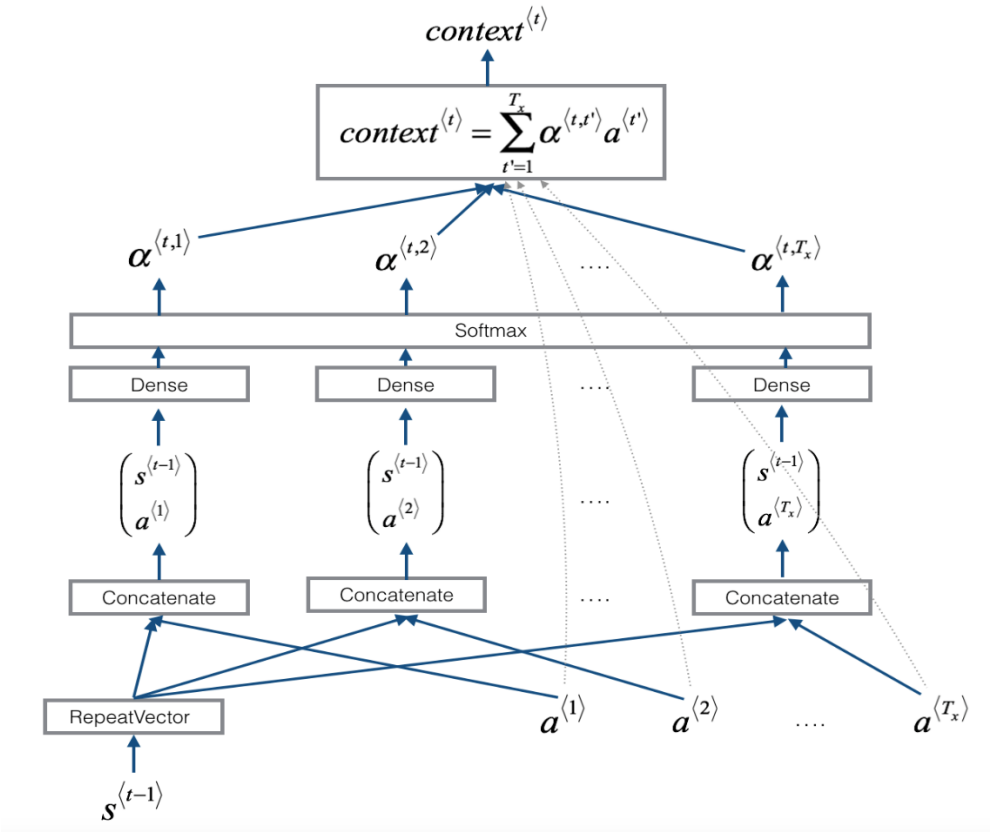
有些事情我还没说清楚，但是在下个视频中，我会讲解一些细节，比如如何准确定义上下文，还有第三个词的上下文，是否真的需要去注意句子中的周围的词。这里要用到的公式以及如何计算这些注意力权重（**these attention weights**），将会在下个视频中讲解到。在下个视频中你会看到$a^{<3,t>}$，即当你尝试去生成第三个词，应该是 **l'Afique**，就得到了右边这个输出，这个 **RNN** 步骤应该要花注意力在t时的法语词上，这取决于在t时的双向 **RNN** 的激活值。那么它应该是取决于第四个激活值，它会取决于上一步的状态，它会取决于$S^{<2>}$。然后这些一起影响你应该花多少注意在输入的法语句子的某个词上面。我们会在下个视频中讲解这些细节。但是直观来想就是 **RNN** 向前进一次生成一个词，在每一步直到最终生成可能是**<EOS>**。这些是注意力权重，即$a^{<t,t>}$告诉你，当你尝试生成第t个英文词，它应该花多少注意力在第t个法语词上面。当生成一个特定的英文词时，这允许它在每个时间步去看周围词距内的法语词要花多少注意力。

我希望这个视频传递了关于注意力模型的一些直观的东西。我们现在可能对算法的运

行有了大概的感觉，让我们进入到下个视频中，看看具体的细节。

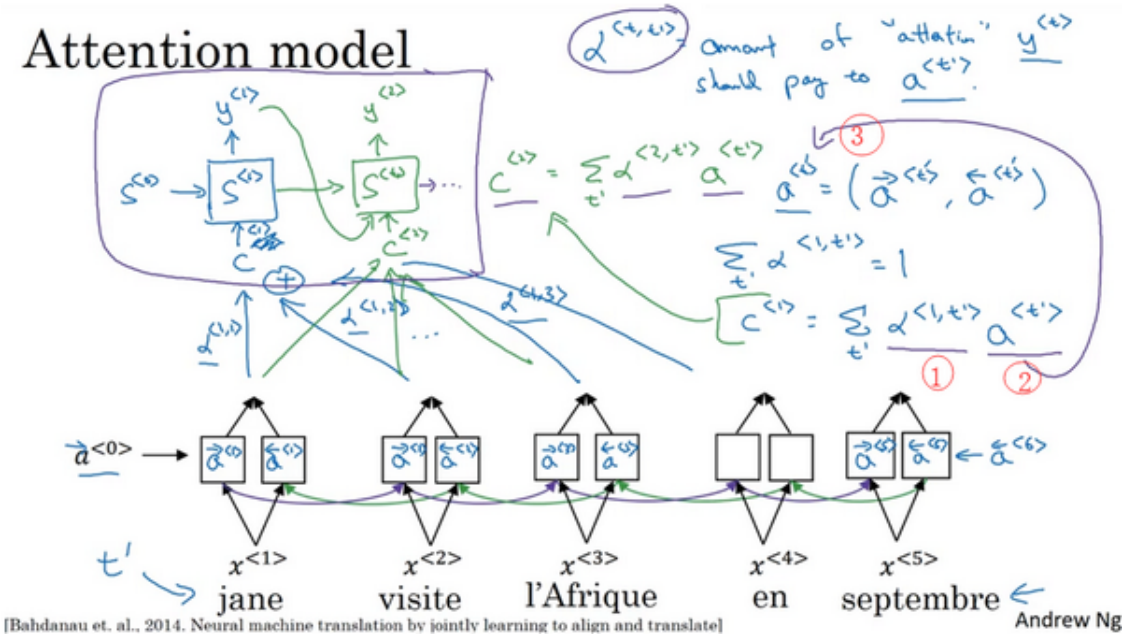
注意力模型参考：




$context^{\langle t \rangle}$
$context^{\langle t \rangle} = \sum_{t'=1}^{T_x} \alpha^{\langle t,t' \rangle} a^{\langle t' \rangle}$
$\alpha^{\langle t,1 \rangle}$
$\alpha^{\langle t,2 \rangle}$
$\alpha^{\langle t,T_x \rangle}$
Softmax
Dense
Dense
Dense
$s^{\langle t-1 \rangle}$
$a^{\langle 1 \rangle}$
$s^{\langle t-1 \rangle}$
$a^{\langle 2 \rangle}$
$s^{\langle t-1 \rangle}$
$a^{\langle T_x \rangle}$
Concatenate
Concatenate
Concatenate
RepeatVector
$a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
$a^{\langle T_x \rangle}$
$s^{\langle t-1 \rangle}$

3.8 注意力模型（Attention Model）

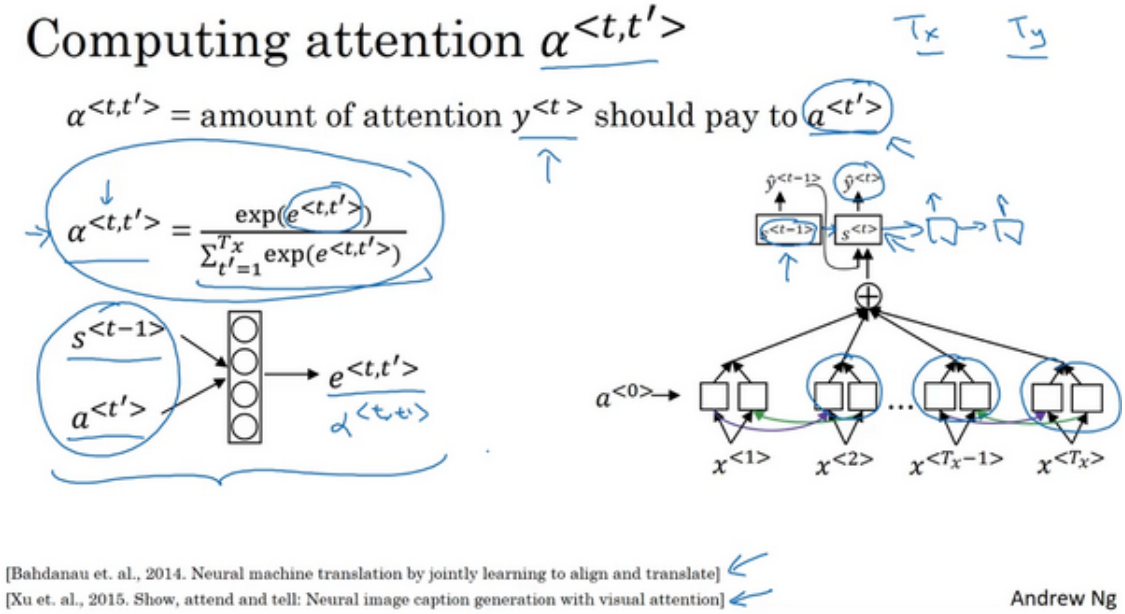
在上个视频中你已经见到了,注意力模型如何让一个神经网络只注意到一部分的输入句子。当它在生成句子的时候，更像人类翻译。让我们把这些想法转化成确切的式子，来实现注意力模型。



[Bahdanau et. al., 2014. Neural machine translation by jointly learning to align and translate]

跟上个视频一样，我们先假定有一个输入句子，并使用双向的 **RNN**，或者双向的 **GRU** 或者双向的 **LSTM**，去计算每个词的特征。实际上 **GRU** 和 **LSTM** 经常应用于这个，可能 **LSTM** 更经常一点。对于前向传播（**the forward occurrence**），你有第一个时间步的前向传播的激活值（**a forward occurrence first time step**），第一个时间步后向传播的激活值，后向的激活值，以此类推。他们一共向前了五个时间步，也向后了五个时间步，技术上我们把这里设置为 0。我们也可以后向传播 6 次，设一个都是 0 的因子，实际上就是个都是 0 的因子。为了简化每个时间步的记号，即使你在双向 RNN 已经计算了前向的特征值和后向的特征值，我就用$a^{<t>}$来一起表示这些联系。所以$a^{<t>}$就是时间步t上的特征向量。但是为了保持记号的一致性，我们用第二个，也就是t'，实际上我将用t'来索引法语句子里面的词。接下来我们只进行前向计算，就是说这是个单向的 **RNN**，用状态S表示生成翻译。所以第一个时间步，它应该生成$y^{<1>}$，当你输入上下文C的时候就会这样，如果你想用时间来索引它，你可以写$C^{<1>}$，但有时候我就写个C，就是没有上标的C，这个会取决于注意力参数，即$a^{<1,1>}$，$a^{<1,2>}$以此类推，告诉我们应该花多少注意力。同样的，这个a参数告诉我们上下文有多少取决于我们得到的特征，或者我们从不同时间步中得到的激活值。所以我们定义上下文的方式实际上来源

于被注意力权重加权的不同时间步中的特征值。于是更公式化的注意力权重将会满足非负的条件，所以这就是个 0 或正数，它们加起来等于 1。我们等会会见到我们如何确保这个成立，我们将会有上下文，或者说在$t = 1$时的上下文，我会经常省略上标，这就会变成对t'的求和。这个权重的所有的t'值，加上这些激活值。所以这里的这项（上图编号 1 所示）就是注意力权重，这里的这项（上图编号 2）来自于这里（上图编号 3），于是$a^{<t,t'>}$就是$y^{<t>}$应该在t'时花在a上注意力的数量。换句话来说，当你在t处生成输出词，你应该花多少注意力在第t'个输入词上面，这是生成输出的其中一步。然后下一个时间步，你会生成第二个输出。于是相似的，你现在有了一个新的注意力权重集，再找到一个新的方式将它们相加，这就产生了一个新的上下文，这个也是输入，且允许你生成第二个词。只有现在才用这种方式相加，它会变成第二个时间步的上下文。即对t'的$a^{<2,t'>}$进行求和，于是使用这些上下文向量，$C^{<1>}$写到这里，$C^{<2>}$也同理。这里的神经网络看起来很像相当标准的 RNN 序列，这里有着上下文向量作为输出，我们可以一次一个词地生成翻译，我们也定义了如何通过这些注意力权重和输入句子的特征值来计算上下文向量。剩下唯一要做的事情就是定义如何计算这些注意力权重。让我们下张幻灯片看看。

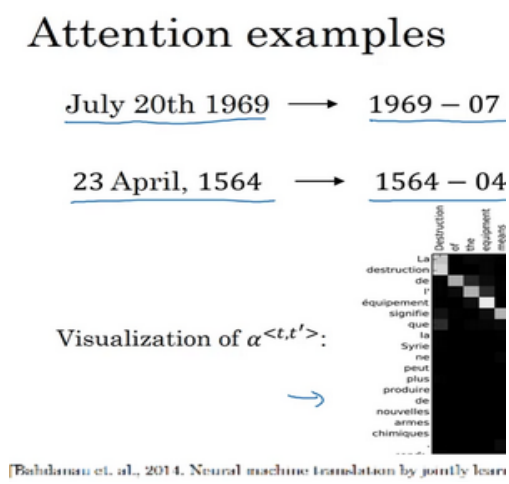


回忆一下$a^{<t,t'>}$，是你应该花费在$a^{<t'>}$上的注意力的数量，当你尝试去生成第t个输出的翻译词，让我们先把式子写下来，再讨论它是怎么来的。这个式子你可以用来计算$a^{<t,t'>}$，在此之前我们要先计算$e^{<t,t'>}$，关键要用 **softmax**，来确保这些权重加起来等于 1。如果你对t'求和，比如每一个固定的t值，这些加起来等于 1。如果你对t'求和，然后优先使用 **softmax**，确保这些值加起来等于 1。

现在我们如何计算这些e项，一种我们可以用的方式是用下面这样的小的神经网络，于

是$s^{<t-1>}$就是神经网络在上个时间步的状态，于是这里我们有一个神经网络，如果你想要生成$y^{<t>}$，那么$s^{<t-1>}$就是上一时间步的隐藏状态，即$s^{<t>}$。这是给小神经网络的其中一个输入，也就是在神经网络中的一个隐藏层，因为你需要经常计算它们，然后$a^{<t'>}$，即上个时间步的的特征是另一个输入。直观来想就是，如果你想要决定要花多少注意力在t'的激活值上。于是，似乎它会很大程度上取决于你上一个时间步的的隐藏状态的激活值。你还没有当前状态的激活值，因为上下文会输入到这里，所以你还没计算出来，但是看看你生成上一个翻译的 RNN 的隐藏状态，然后对于每一个位置，每一个词都看向他们的特征值，这看起来很自然，即$a^{<t,t'>}$和$e^{<t,t'>}$应该取决于这两个量。但是我们不知道具体函数是什么，所以我们可以做的事情就是训练一个很小的神经网络，去学习这个函数到底是什么。相信反向传播算法，相信梯度下降算法学到一个正确的函数。这表示，如果你应用这整个的模型，然后用梯度下降来训练它，这是可行的。这个小型的神经网络做了一件相当棒的事情，告诉你$y^{<t>}$应该花多少注意力在$a^{<t>}$上面，然后这个式子确保注意力权重加起来等于 1，于是当你持续地一次生成一个词，这个神经网络实际上会花注意力在右边的这个输入句子上，它会完全自动的通过梯度下降来学习。

这个算法的一个缺点就是它要花费三次方的时间，就是说这个算法的复杂是$O(n3)$的，如果你有T_x个输入单词和T_y个输出单词，于是注意力参数的总数就会是$T_x \times T_y$，所以这个算法有着三次方的消耗。但是在机器翻译的应用上，输入和输出的句子一般不会太长，可能三次方的消耗是可以接受，但也有很多研究工作，尝试去减少这样的消耗。那么讲解注意想法在机器翻译中的应用，就到此为止了。虽然没有讲到太多的细节，但这个想法也被应用到了其他的很多问题中去了，比如图片加标题（**image captioning**），图片加标题就是看一张图，写下这张图的标题。底下的这篇论文来源于 **Kevin Chu**，**Jimmy Barr**, **Ryan Kiros**, **Kelvin Shaw**, **Aaron Korver**, **Russell Zarkutnov**, **Virta Zemo**, 和 **Andrew Benjo**。他们也显示了你可以有一个很相似的结构看图片，然后，当你在写图片标题的时候，一次只花注意力在一部分的图片上面。如果你感兴趣，那么我鼓励你，也去看看这篇论文，做一些编程练习。



因为机器翻译是一个非常复杂的问题，在之前的练习中，你应用了注意力，在日期标准化的问题（**the date normalization problem**）上面，问题输入了像这样的一个日期，这个日期实际上是阿波罗登月的日期，把它标准化成标准的形式，或者这样的日期。用一个序列的神经网络，即序列模型去标准化到这样的形式，这个日期实际上是威廉·莎士比亚的生日。一般认为是这个日期正如你之前练习中见到的，你可以训练一个神经网络，输入任何形式的日期，生成标准化的日期形式。其他可以做的有意思的事情是看看可视化的注意力权重（**the visualizations of the attention weights**）。这个一个机器翻译的例子，这里被画上了不同的颜色，不同注意力权重的大小，我不想在这上面花太多时间，但是你可以发现，对应的输入输出词，你会发现注意力权重，会变高，因此这显示了当它生成特定的输出词时通常会花注意力在输入的正确的词上面，包括学习花注意在哪。 在注意力模型中，使用反向传播时， 什么时候学习完成。

这就是注意力模型，在深度学习中真的是个非常强大的想法。在本周的编程练习中，我希望你可以享受自己应用它的过程。