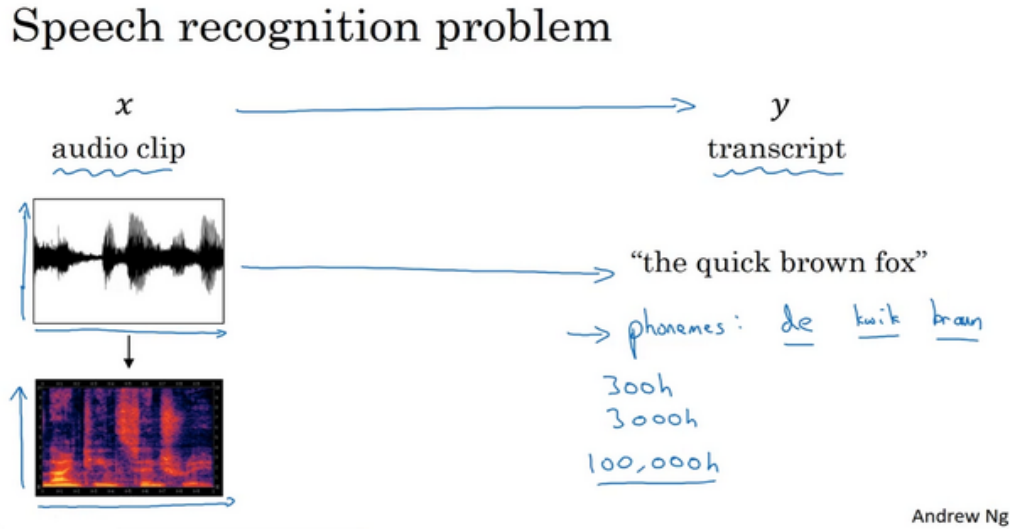
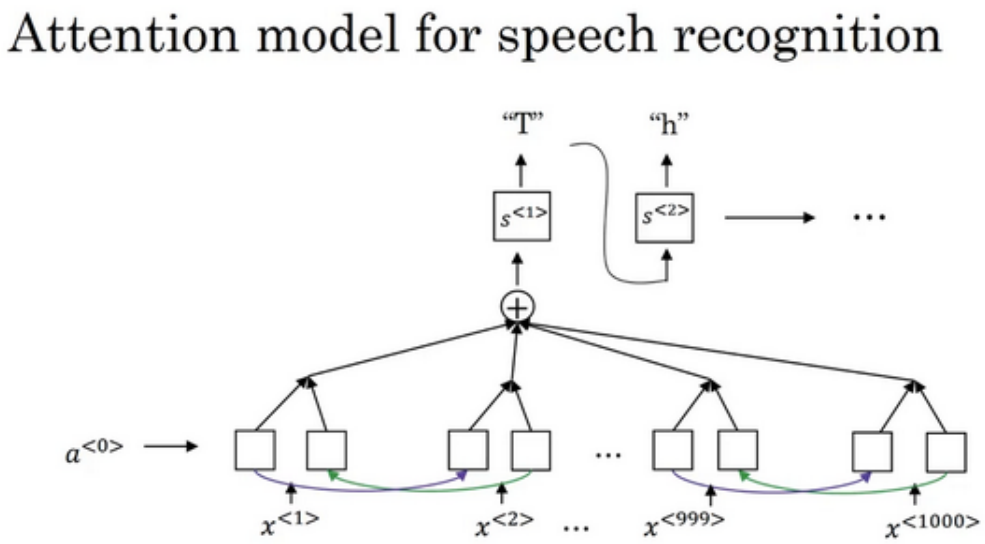
3.9 语音识别（Speech recognition）

现今，最令人振奋的发展之一，就是 **seq2seq** 模型（**sequence-to-sequence models**）在语音识别方面准确性有了很大的提升。这门课程已经接近尾声，现在我想通过剩下几节视频，来告诉你们，**seq2seq** 模型是如何应用于音频数据的（**audio data**），比如语音（**the speech**）。

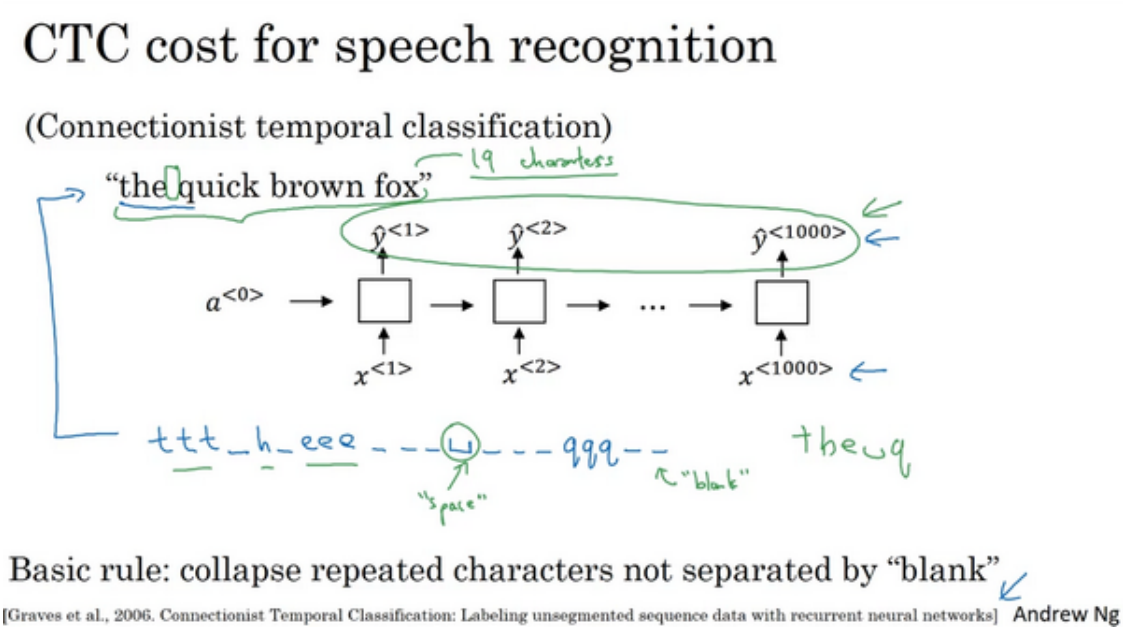


什么是语音视频问题呢？现在你有一个音频片段x（**an audio clip,x**），你的任务是自动地生成文本y。现在有一个音频片段，画出来是这样，该图的横轴是时间。一个麦克风的作用是测量出微小的气压变化，现在你之所以能听到我的声音，是因为你的耳朵能够探测到这些微小的气压变化，它可能是由你的扬声器或者耳机产生的，也就是像图上这样的音频片段，气压随着时间而变化。假如这个我说的音频片段的内容是：**"the quick brown fox"**(敏捷的棕色狐狸)，这时我们希望一个语音识别算法（**a speech recognition algorithm**），通过输入这段音频，然后输出音频的文本内容。考虑到人的耳朵并不会处理声音的原始波形，而是通过一种特殊的物理结构来测量这些，不同频率和强度的声波。音频数据的常见预处理步骤，就是运行这个原始的音频片段，然后生成一个声谱图（**a spectrogram**），就像这样。同样地，横轴是时间，纵轴是声音的频率（**frequencies**），而图中不同的颜色，显示了声波能量的大小（**the amount of energy**），也就是在不同的时间和频率上这些声音有多大。通过这样的声谱图，或者你可能还听过人们谈到过伪空白输出（**the false blank outputs**），也经常应用于预处理步骤，也就是在音频被输入到学习算法之前，而人耳所做的计算和这个预处理过程非常相似。语音识别方面，最令人振奋的趋势之一就是曾经有一段时间，语音识别系统是用音位（**phonemes**）来构建的，也就是人工设计的基本单元（**hand-engineered basic units of cells**），

如果用音位来表示"**the quick brown fox**"，我这里稍微简化一些，"**the**"含有"**th**"和"**e**"的音，而"**quick**"有"**k**" "**w**" "**i**" "**k**"的音，语音学家过去把这些音作为声音的基本单元写下来，把这些语音分解成这些基本的声音单元，而"**brown**"不是一个很正式的音位，因为它的音写起来比较复杂，不过语音学家（**linguists**）们认为用这些基本的音位单元（**basic units of sound called phonemes**）来表示音频（**audio**），是做语音识别最好的办法。不过在 **end-to-end** 模型中，我们发现这种音位表示法（**phonemes representations**）已经不再必要了，而是可以构建一个系统，通过向系统中输入音频片段（**audio clip**），然后直接输出音频的文本（**a transcript**），而不需要使用这种人工设计的表示方法。使这种方法成为可能的一件事就是用一个很大的数据集，所以语音识别的研究数据集可能长达 300 个小时，在学术界，甚至 3000 小时的文本音频数据集，都被认为是合理的大小。大量的研究，大量的论文所使用的数据集中，有几千种不同的声音，而且，最好的商业系统现在已经训练了超过 1 万个小时的数据，甚至 10 万个小时，并且它还会继续变得更大。在文本音频数据集中（**Transcribe audio data sets**）同时包含x和y，通过深度学习算法大大推进了语音识别的进程。那么，如何建立一个语音识别系统呢？



在上一节视频中，我们谈到了注意力模型，所以，一件你能做的事就是在横轴上，也就是在输入音频的不同时间帧上，你可以用一个注意力模型，来输出文本描述，如"**the quick brown fox**"，或者其他语音内容。



还有一种效果也不错的方法，就是用 **CTC** 损失函数（**CTC cost**）来做语音识别。**CTC** 就是 **Connectionist Temporal Classification**，它是由 **Alex Graves**、**Santiago Fernandes, Faustino Gomez**、和 **Jürgen Schmidhuber** 提出的。（**Graves A, Gomez F. Connectionist temporal classification:labelling unsegmented sequence data with recurrent neural networks[C]// International Conference on Machine Learning. ACM, 2006:369-376.**）

算法思想如下:

假设语音片段内容是某人说："**the quick brown fox**"，这时我们使用一个新的网络，结构像这个样子，这里输入x和输出y的数量都是一样的，因为我在这里画的，只是一个简单的单向 **RNN** 结构。然而在实际中，它有可能是双向的 **LSTM** 结构，或者双向的 **GIU** 结构，并且通常是很深的模型。但注意一下这里时间步的数量，它非常地大。在语音识别中，通常输入的时间步数量（**the number of input time steps**）要比输出的时间步的数量（**the number of output time steps**）多出很多。举个例子，比如你有一段 10 秒的音频，并且特征（**features**）是 100 赫兹的，即每秒有 100 个样本，于是这段 10 秒的音频片段就会有 1000 个输入，就是简单地用 100 赫兹乘上 10 秒。所以有 1000 个输入，但可能你的输出就没有 1000 个字母了，或者说没有 1000 个字符。这时要怎么办呢？**CTC** 损失函数允许 **RNN** 生成这样的输出：**ttt**，这是一个特殊的字符，叫做空白符，我们这里用下划线表示，这句话开头的音可表示为 **h_eee___**，然后这里可能有个空格，我们用这个来表示空格，之后是**___qqq__**，这样的输出也被看做是正确的输出。下面这段输出对应的是"**the q**"。**CTC 损失函数的一个基本规则是将空白符之间的重复的字符折叠起来，再说清楚一些，我这里用下划线来表示这个特殊的**

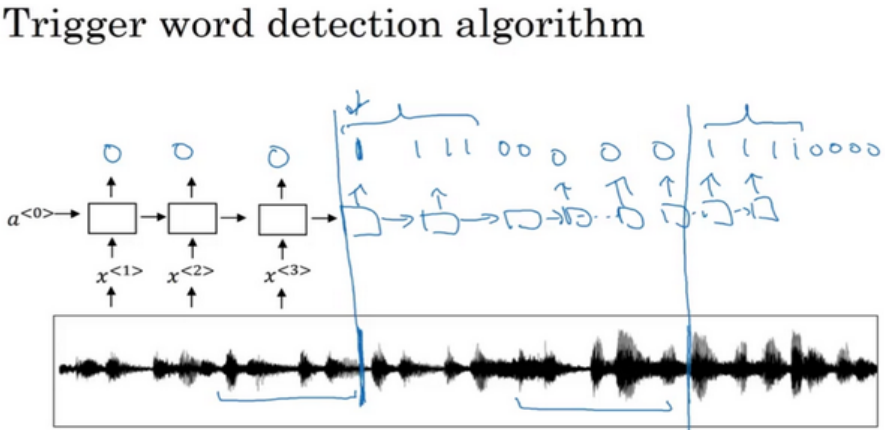
空白符（a special blank character），它和空格（the space character）是不一样的。所以 **the** 和 **quick** 之间有一个空格符，所以我要输出一个空格，通过把用空白符所分割的重复的字符折叠起来，然后我们就可以把这段序列折叠成"**the q**"。这样一来你的神经网络因为有很多这种重复的字符，和很多插入在其中的空白符（**blank characters**），所以最后我们得到的文本会短上很多。于是这句"**the quick brown fox**"包括空格一共有 19 个字符，在这样的情况下，通过允许神经网络有重复的字符和插入空白符使得它能强制输出 1000 个字符，甚至你可以输出 1000 个y值来表示这段 19 个字符长的输出。这篇论文来自于 **Alex Grace** 以及刚才提到的那些人。我所参与的深度语音识别系统项目就使用这种思想来构建有效的语音识别系统。

希望这能给你一个粗略的理解，理解语音识别模型是如何工作的：注意力模型是如何工作的，以及 **CTC** 模型是如何工作的，以及这两种不同的构建这些系统的方法。现今，在生产技术中，构建一个有效语音识别系统，是一项相当重要的工作，并且它需要很大的数据集，下节视频我想做的是告诉你如何构建一个触发字检测系统（**a rigger word detection system**），其中的关键字检测系统（**keyword detection system**）将会更加简单，它可以通过一个更简洁的数量更合理的数据来完成。所以我们下节课再见。

3.10 触发字检测（Trigger Word Detection）

现在你已经学习了很多关于深度学习和序列模型的内容，于是我们可以真正去简便地描绘出一个触发字系统（**a trigger word system**），就像上节视频中你看到的那样。随着语音识别的发展，越来越多的设备可以通过你的声音来唤醒，这有时被叫做触发字检测系统（**rigger word detection systems**）。我们来看一看如何建立一个触发字系统。

触发字系统的例子包括 **Amazon echo**，它通过单词 **Alexa** 唤醒；还有百度 **DuerOS** 设备，通过"小度你好"来唤醒；苹果的 **Siri** 用 **Hey Siri** 来唤醒；**Google Home** 使用 **Okay Google** 来唤醒，这就是触发字检测系统。假如你在卧室中，有一台 **Amazon echo**，你可以在卧室中简单说一句: **Alexa**, 现在几点了?就能唤醒这个设备。它将会被单词"**Alexa**"唤醒，并回答你的询问。如果你能建立一个触发字检测系统，也许你就能让你的电脑通过你的声音来执行某些事，我有个朋友也在做一种用触发字来打开的特殊的灯，这是个很有趣的项目。但我想教会你的，是如何构建一个触发字检测系统。



有关于触发字检测系统的文献，还处于发展阶段。对于触发字检测，最好的算法是什么，目前还没有一个广泛的定论。我这里就简单向你介绍一个你能够使用的算法好了。现在有一

个这样的 **RNN** 结构，**我们要做的就是把一个音频片段（an audio clip）计算出它的声谱图特征（spectrogram features）得到特征向量**$x^{<1>}$, $x^{<2>}$, $x^{<3>}$..，然后把它放到 **RNN** 中，最后要做的，就是定义我们的目标标签y。假如音频片段中的这一点是某人刚刚说完一个触发字，比如"**Alexa**"，或者"小度你好" 或者"**Okay Google**"，那么在这一点之前，你就可以在训练集中把目标标签都设为 0，然后在这个点之后把目标标签设为 1。假如在一段时间之后，触发字又被说了一次，比如是在这个点说的，那么就可以再次在这个点之后把目标标签设为 1。这样的标签方案对于 **RNN** 来说是可行的，并且确实运行得非常不错。不过该算法一个明显的缺点就是它构建了一个很不平衡的训练集（**a very imbalanced training set**），0 的数量比 1 多太多了。

这里还有一个解决方法，虽然听起来有点简单粗暴，但确实能使其变得更容易训练。比起只在一个时间步上去输出 1，其实你可以在输出变回 0 之前，多次输出 1，或说在固定的一段时间内输出多个 1。这样的话，就稍微提高了 1 与 0 的比例，这确实有些简单粗暴。在音频片段中，触发字刚被说完之后，就把多个目标标签设为 1，这里触发字又被说了一次。说完以后，又让 **RNN** 去输出 1。在之后的编程练习中，你可以进行更多这样的操作，我想你应该会对自己学会了这么多东西而感到自豪。我们仅仅用了一张幻灯片来描述这种复杂的触发字检测系统。在这个基础上，希望你能够实现一个能有效地让你能够检测出触发字的算法，不过在编程练习中你可以看到更多的学习内容。这就是触发字检测，希望你能对自己感到自豪。因为你已经学了这么多深度学习的内容，现在你可以只用几分钟时间，就能用一张幻灯片来描述触发字能够实现它，并让它发挥作用。你甚至可能在你的家里用触发字系统做一些有趣的事情，比如打开或关闭电器，或者可以改造你的电脑，使得你或者其他人可以用触发字来操作它。

这是深度学习课程最后一个技术视频，所以总结一下我们对序列模型的学习。我们学了 **RNN**，包括 **GRU** 和 **LSTM**，然后在上一周我们学了词嵌入（**word embeddings**），以及它们如何学习词汇的表达（**how they learn representations of words**）。在这周还学了注意力模型（**the attention model**）以及如何使用它来处理音频数据（**audio data**）。希望你在编程练习中实现这些思想的时候，能够体会到诸多乐趣。接下来我们来看最后一个视频。