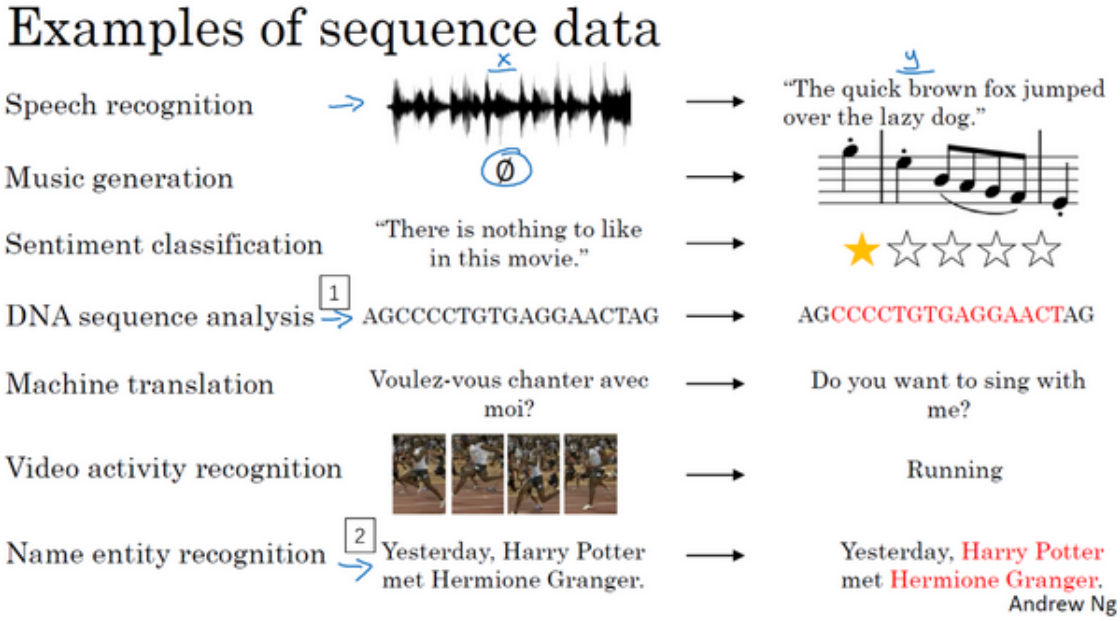
1.1 为什么选择序列模型？（Why Sequence Models?）

在本课程中你将学会序列模型，它是深度学习中最令人激动的内容之一。循环神经网络（**RNN**）之类的模型在**语音识别、自然语言处理**和其他领域中引起变革。在本节课中，你将学会如何自行创建这些模型。我们先看一些例子，这些例子都有效使用了序列模型。



在进行语音识别时，给定了一个输入音频片段 x，并要求输出对应的文字记录 y。这个例子里输入和输出数据都是序列模型，因为 x是一个按时播放的音频片段，输出 y是一系列单词。所以之后将要学到的一些序列模型，如循环神经网络等等在语音识别方面是非常有用的。

音乐生成问题是使用序列数据的另一个例子，在这个例子中，只有输出数据 y是序列，而输入数据可以是空集，也可以是个单一的整数，这个数可能指代你想要生成的音乐风格，也可能是你想要生成的那首曲子的头几个音符。输入的 x可以是空的，或者就是个数字，然后输出序列y。

在处理情感分类时，输入数据x是序列，你会得到类似这样的输入：**“There is nothing to like in this movie.”**，你认为这句评论对应几星？

系列模型在 **DNA** 序列分析中也十分有用，你的 **DNA** 可以用 **A**、**C**、**G**、**T** 四个字母来表示。所以给定一段 **DNA** 序列，你能够标记出哪部分是匹配某种蛋白质的吗？

在机器翻译过程中，你会得到这样的输入句：**“Voulez-vou chante avecmoi?”**（法语：要和我一起唱么？），然后要求你输出另一种语言的翻译结果。

在进行视频行为识别时，你可能会得到一系列视频帧，然后要求你识别其中的行为。

在进行命名实体识别时，可能会给定一个句子要你识别出句中的人名。

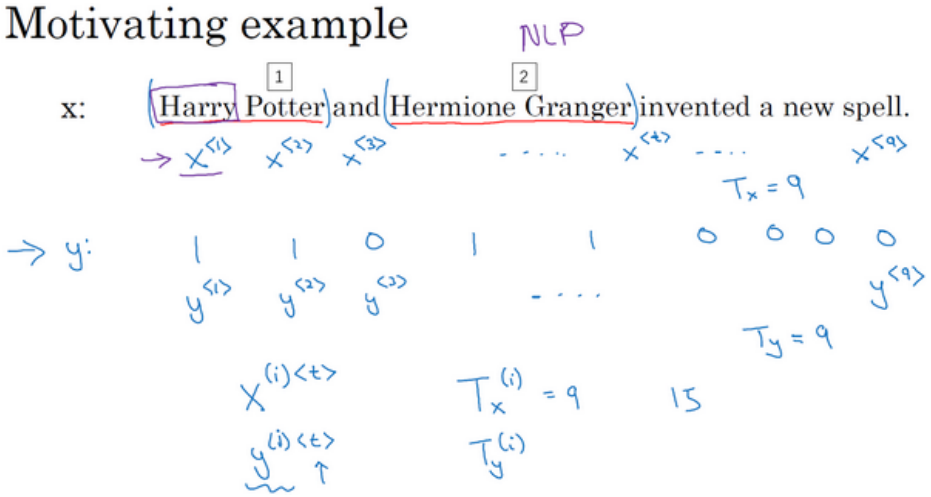
所以这些问题都可以被称作**使用标签数据 (x, y)作为训练集的监督学习**。但从这一系列例子中你可以看出序列问题有很多不同类型。有些问题里，输入数据 x和输出数据y都是序列，但就算在那种情况下，x和y有时也会不一样长。或者像上图编号 1 所示和上图编号 2 的x和y有相同的数据长度。在另一些问题里，只有 x或者只有y是序列。

所以在本节我们学到适用于不同情况的序列模型，下节中我们会定义一些定义序列问题要用到的符号。

1.2 数学符号（Notation）

本节先从定义符号开始一步步构建序列模型。

比如说你想要建立一个序列模型，它的输入语句是这样的：“**Harry Potter and Herminoe Granger invented a new spell.**”，(这些人名都是出自于 **J.K.Rowling** 笔下的系列小说 **Harry Potter**)。假如你想要建立一个能够自动识别句中人名位置的序列模型，那么这就是一个命名实体识别问题，这常用于搜索引擎，比如说索引过去 24 小时内所有新闻报道提及的人名，用这种方式就能够恰当地进行索引。命名实体识别系统可以用来查找不同类型的文本中的人名、公司名、时间、地点、国家名和货币名等等。



现在给定这样的输入数据x，假如你想要一个序列模型输出y，使得输入的每个单词都对应一个输出值，同时这个y能够表明输入的单词是否是人名的一部分。技术上来说这也许不是最好的输出形式，还有更加复杂的输出形式，它不仅能够表明输入词是否是人名的一部分，它还能够告诉你这个人名在这个句子里从哪里开始到哪里结束。比如 **Harry Potter**（上图编号 1 所示）、**Hermione Granger**（上图标号 2 所示）。

更简单的那种输出形式:

这个输入数据是 9 个单词组成的序列，所以最终我们会有 9 个特征集和来表示这 9 个单词，并按序列中的位置进行索引，$x^{<1>}$、$x^{<2>}$、$x^{<3>}$等等一直到$x^{<9>}$来索引不同的位置，我将用$x^{<t>}$来索引这个序列的中间位置。***t*意味着它们是时序序列**，但不论是否是时序序列，我们都将用t来索引序列中的位置。

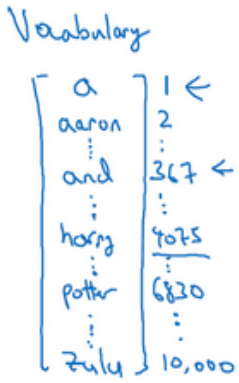
输出数据也是一样，我们还是用$y^{<1>}$、$y^{<2>}$、$y^{<3>}$等等一直到$y^{<9>}$来表示输出数据。同时我们用T_x来表示输入序列的长度，这个例子中输入是 9 个单词，所以$T_x = 9$。我们用T_y来表示输出序列的长度。在这个例子里$T_x = T_y$，上个视频里你知道T_x和T_y可以有不同的值。

你应该记得我们之前用的符号，我们用$x^{(i)}$来表示第i个训练样本，所以为了指代第t个元素，或者说是训练样本i的序列中第t个元素用$x^{(i)<t>}$这个符号来表示。如果T_x是序列长度，那么你的训练集里不同的训练样本就会有不同的长度，所以$T_x^{(i)}$就代表第i个训练样本的输入序列长度。同样$y^{(i)<t>}$代表第i个训练样本中第t个元素，$T_y^{(i)}$就是第i个训练样本的输出序列的长度。

所以在这个例子中，$T_x^{(i)} = 9$，但如果另一个样本是由 15 个单词组成的句子，那么对于这个训练样本，$T_x^{(i)} = 15$。

既然我们这个例子是 **NLP**，也就是自然语言处理，这是我们初次涉足自然语言处理，一件我们需要事先决定的事是怎样表示一个序列里单独的单词，你会怎样表示像 **Harry** 这样的单词，$x^{<1>}$实际应该是什么？

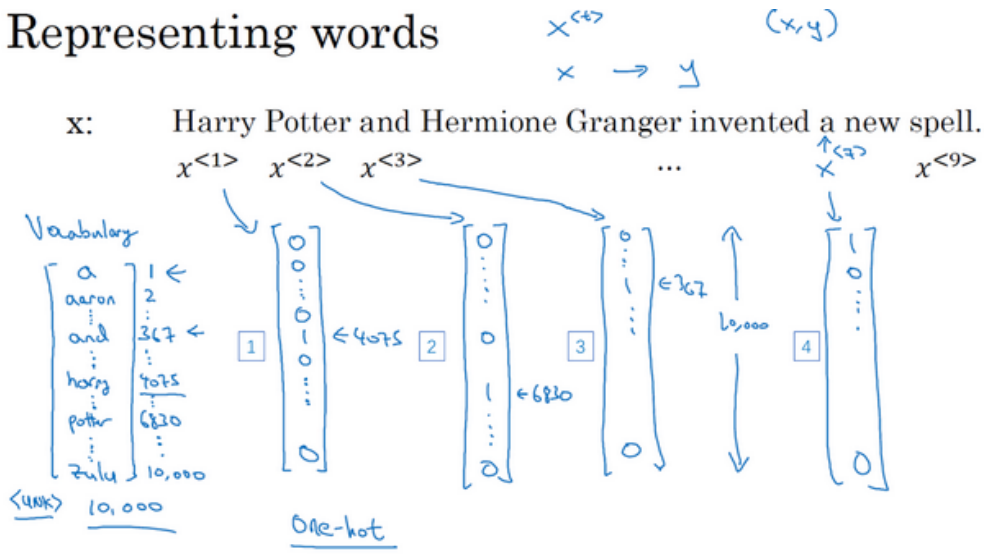
接下来我们讨论一下怎样表示一个句子里单个的词。想要表示一个句子里的单词，第一件事是做一张词表，有时也称为词典，意思是列一列你的表示方法中用到的单词。这个词表（下图所示）中的第一个词是 **a**，也就是说词典中的第一个单词是 **a**，第二个单词是 **Aaron**，然后更下面一些是单词 **and**，再后面你会找到 **Harry**，然后找到 **Potter**，这样一直到最后，词典里最后一个单词可能是 **Zulu**。



因此 **a** 是第一个单词，**Aaron** 是第二个单词，在这个词典里，**and** 出现在 367 这个位置上，**Harry** 是在 4075 这个位置，**Potter** 在 6830，词典里的最后一个单词 **Zulu** 可能是第 10,000 个单词。所以在这个例子中我用了 10,000 个单词大小的词典，这对现代自然语言处理应用来说太小了。对于商业应用来说，或者对于一般规模的商业应用来说 30,000 到 50,000 词大小的词典比较常见，但是 100,000 词的也不是没有，而且有些大型互联网公司会用百万词，甚至更大的词典。许多商业应用用的词典可能是 30,000 词，也可能是 50,000 词。不过我将用 10,000 词大小的词典做说明，因为这是一个很好用的整数。

如果你选定了 10,000 词的词典，构建这个词典的一个方法是遍历你的训练集，并且找

到前 10,000 个常用词，你也可以去浏览一些网络词典，它能告诉你英语里最常用的 10,000 个单词，接下来你可以用 **one-hot 表示法来表示词典里的每个单词**。



举个例子，在这里$x^{<1>}$表示 **Harry** 这个单词，它就是一个第 4075 行是 1，其余值都是 0 的向量（上图编号 1 所示），因为那是 **Harry** 在这个词典里的位置。

同样$x^{<2>}$是个第 6830 行是 1，其余位置都是 0 的向量（上图编号 2 所示）。

and 在词典里排第 367，所以$x^{<3>}$就是第 367 行是 1，其余值都是 0 的向量（上图编号 3 所示）。如果你的词典大小是 10,000 的话，那么这里的每个向量都是 10,000 维的。

因为 **a** 是字典第一个单词，$x^{<7>}$对应 **a**，那么这个向量的第一个位置为 1，其余位置都是 0 的向量（上图编号 4 所示）。

所以这种表示方法中，$x^{<t>}$指代句子里的任意词，它就是个 **one-hot** 向量，因为它只有一个值是 1，其余值都是 0，所以你会有 9 个 **one-hot** 向量来表示这个句中的 9 个单词，目的是用这样的表示方式表示X，用序列模型在X和目标输出Y之间学习建立一个映射。我会把它当作监督学习的问题，我确信会给定带有(x, y)标签的数据。

那么还剩下最后一件事，我们将在之后的视频讨论，如果你遇到了一个不在你词表中的单词，答案就是创建一个新的标记，也就是一个叫做 **Unknow Word** 的伪单词，用**<UNK>**作为标记，来表示不在词表中的单词，我们之后会讨论更多有关这个的内容。