deeplearning.ai

吴恩达 DeepLearning.ai 课程提炼笔记（5-3）序列模型 --- 序列模型和注意力机制

完结撒花！以下为吴恩达老师 DeepLearning.ai 课程项目中，第五部分《序列模型》第三周课程“**序列模型和注意力机制**”关键点的笔记。

---

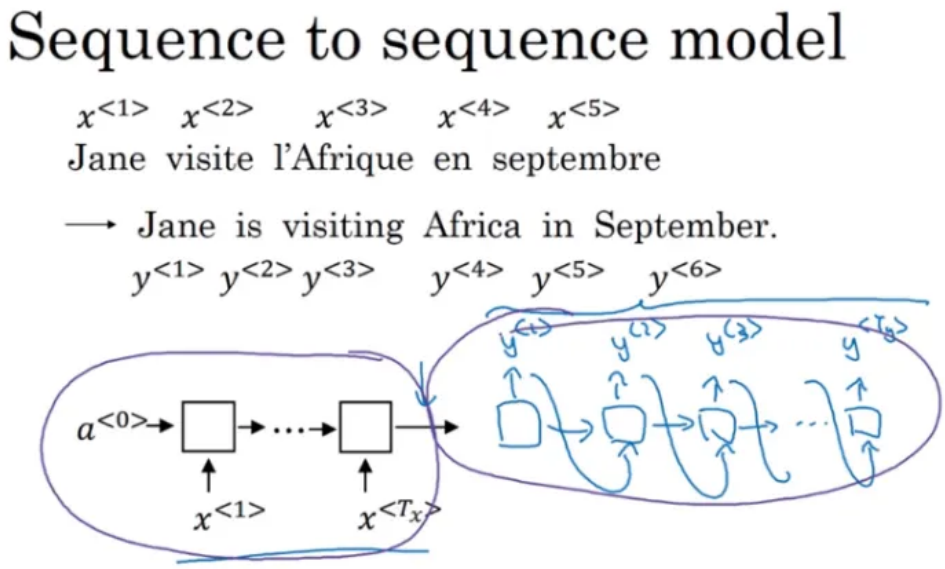
1. 基础模型

sequence to sequence 模型：

sequence to sequence 模型最为常见的就是机器翻译，假如这里我们要将法语翻译成英文：

- 输入：$x^{<1>}$，$x^{<2>}$，$\cdots$，$x^{<T_x>}$；这里每个 $x^{<t>}$ 均为对应法语句子中的每个单词；
- 输出：$y^{<1>}$，$y^{<2>}$，$\cdots$，$y^{<T_y>}$；这里每个 $y^{<t>}$ 均为对应英语句子中的每个单词；
- 网络结构：many-to-many RNN网络结构。

对于机器翻译的序列对序列模型，如果我们拥有大量的句子语料，则可以得到一个很有效的机器翻译模型。模型的前部分使用一个编码网络来对输入的法语句子进行编码，后半部分则使用一个解码网络来生成对应的英文翻译。网络结构如下图所示：



相关论文：

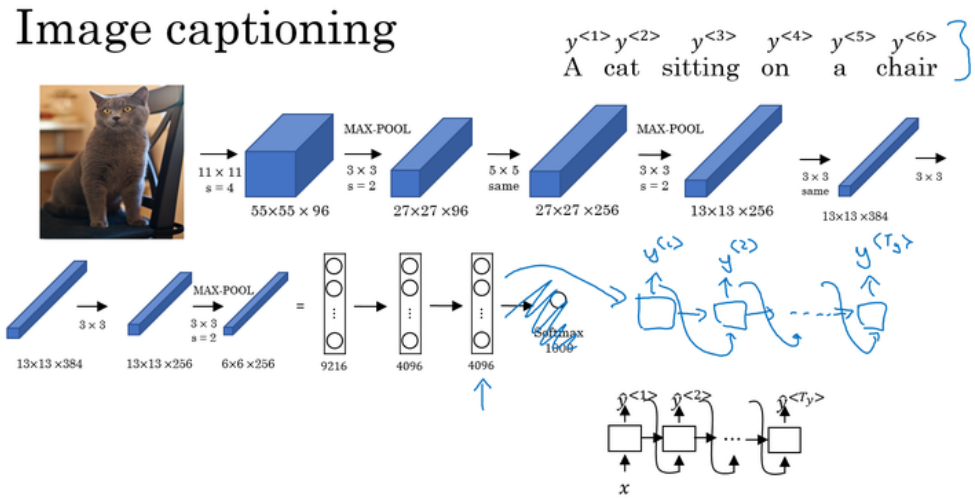
- Sutskever et al., Sequence to sequence learning with neural networks, 2014;
- Cho et al., Learning phrase representation using RNN encoder-decoder for statistical machine translation, 2014;

image to sequence 模型：

与上面的这种编解码类似的还有就是图像描述的应用。

- 输入：图像；
- 输出：描述图像的句子；
- 网络结构：CNN结构学习图像的编码，RNN结构解码输出对应的句子。

对于图像描述的网络结构如下图所示：



相关论文：

- Mao et. al., 2014. Deep captioning with multimodal recurrent neural networks；
- Vinyals et. al., 2014. Show and tell: Neural image caption generator；
- Karpathy and Li, 2015. Deep visual-semantic alignments for generating image descriptions；

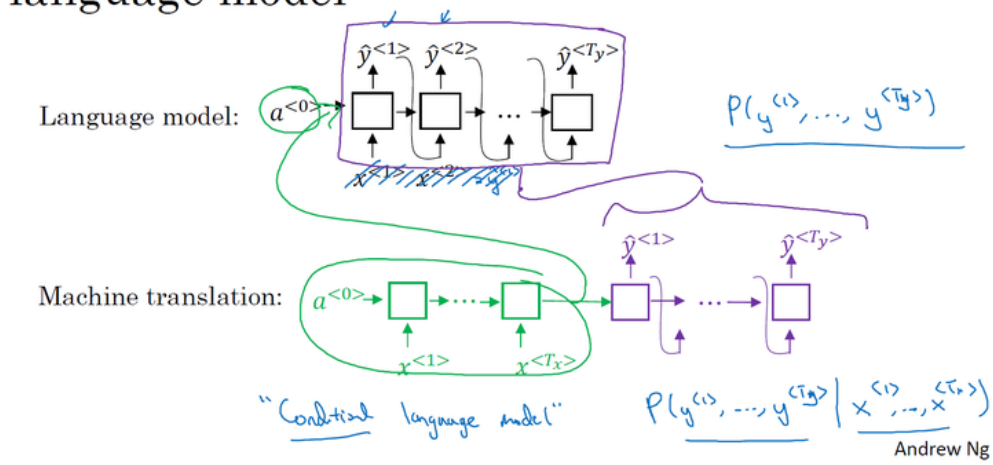
2. 挑选最可能的句子

机器翻译：条件语言模型

对于机器翻译来说和之前几节介绍的语言模型有很大的相似性但也有不同之处。

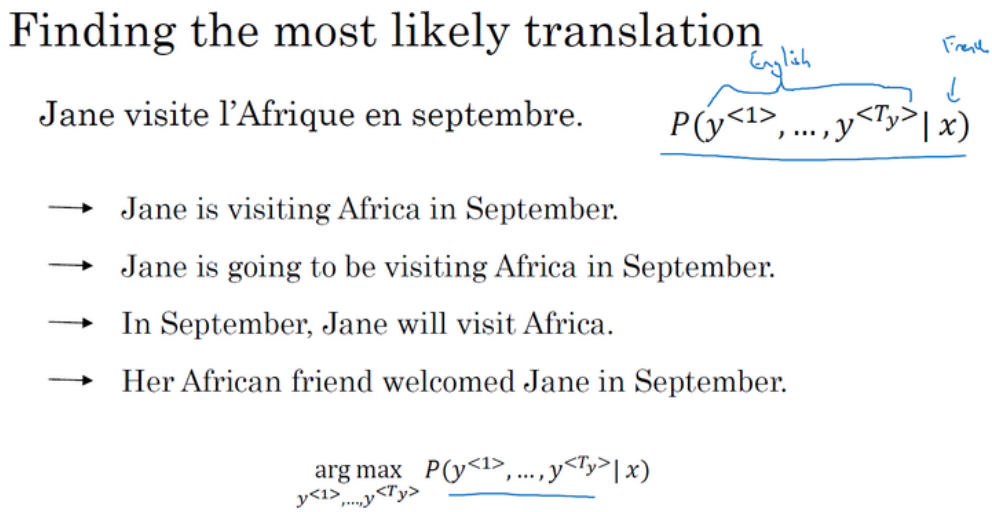
- 在语言模型中，我们通过估计句子的可能性，来生成新的句子。语言模型总是以零向量开始，也就是其第一个时间步的输入可以直接为零向量；
- 在机器翻译中，包含了编码网络和解码网络，其中解码网络的结构与语言模型的结构是相似的。机器翻译以句子中每个单词的一系列向量作为输入，所以相比语言模型来说，机器翻译可以称作**条件语言模型**，其输出的句子概率是相对于输入的条件概率。

二者对比如下图所示：



寻找最大的概率输出：

还是以法语翻译英语为例，通过输入的法语句子，模型将会告诉我们各种英文句子输出所对应的可能性，如下图中的句子所示。



$$\arg\max_{y^{<1>},\dots,y^{<T_y>}} P(y^{<1>},\dots,y^{<T_y>}|x)$$

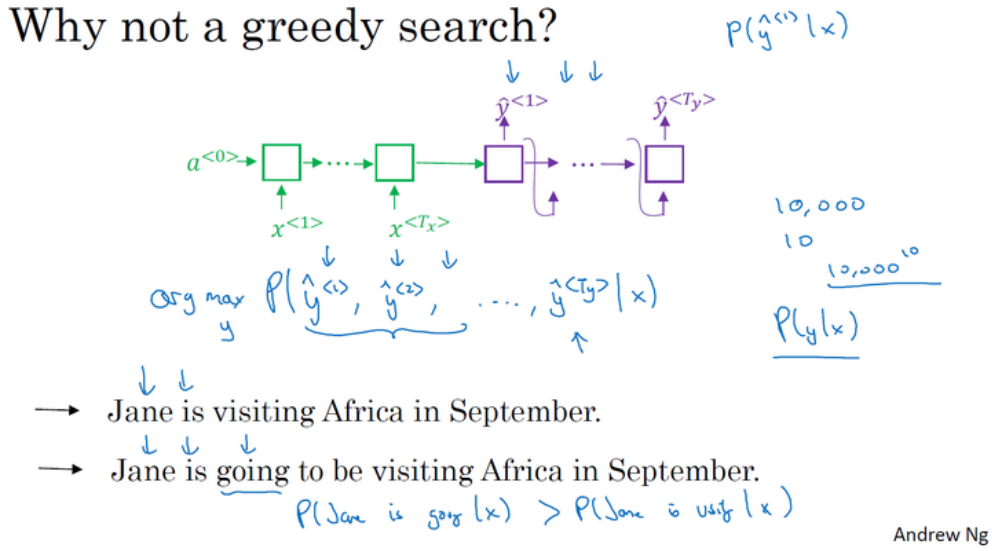
对于各种可能的翻译结果，我们并不是要从得到的分布中进行随机取样，而是要找到一个使得条件概率最大化的英文句子作为输出。所以在设计机器翻译模型的时候，一个重要的步骤就是设计一个合适的算法，找到使得条件概率最大化的的结果。目前最通用的算法就是：束搜索（Beam Search）。

不使用贪心搜索的原因：

对于我们的机器翻译模型来说，使用贪心搜索算法，在生成第一个词的分布后，贪心搜索会根据我们的条件语言模型挑选出最有可能输出的第一个词语，然后再挑选出第二个最有可能的输出词语，依次给出所有的输出。

但是对于我们建立的机器翻译模型来说，我们真正需要的是通过模型一次性地挑选出整个输出序列：$y^{<1>}$，$y^{<2>}$，$\cdots$，$y^{<T_y>}$，来使得整体的概率最大化。所以对于贪心搜索来说，这种方法对于机器翻译来说是不可行的。

另外对于贪心搜索算法来说，我们的单词库中有成百到千万的词汇，去计算每一种单词的组合的可能性是不可行的。所以我们使用近似的搜索办法，使得条件概率最大化或者近似最大化的句子，而不是通过单词去实现，虽然不能保证我们得到的就是条件概率最大化的结果，但是往往这已经足够了。

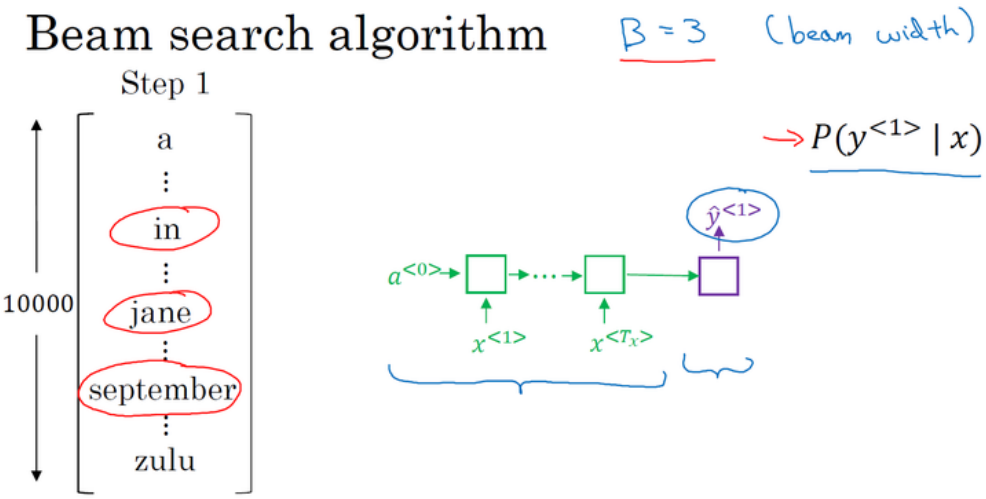


3. 集束搜索（Beam search）

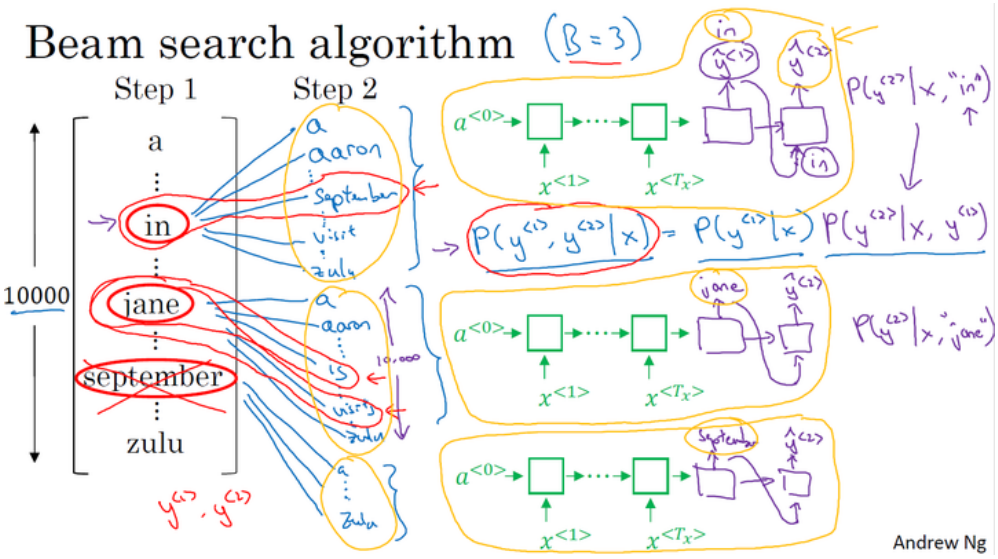
Beam search 算法：

这里我们还是以法语翻译成英语的机器翻译为例：

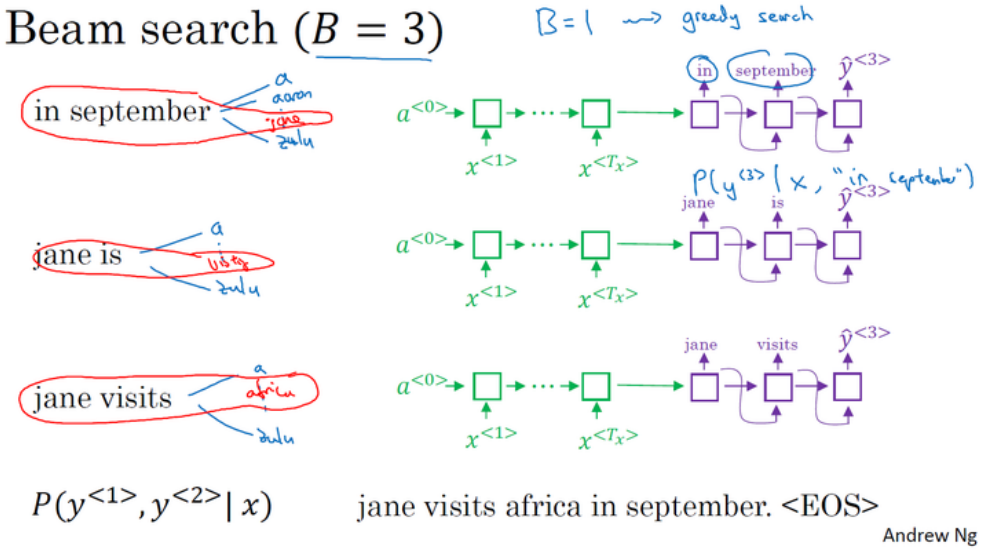
- Step 1：对于我们的词汇表，我们将法语句子输入到编码网络中得到句子的编码，通过一个softmax层计算各个单词（词汇表中的所有单词）输出的概率值，通过设置集束宽度（beam width）的大小如3，我们则取前3个最大输出概率的单词，并保存起来。



- Step 2：在第一步中得到的集束宽度的单词数，我们分别对第一步得到的每一个单词计算其与单词表中的所有单词组成词对的概率。并与第一步的概率相乘，得到第一和第二两个词对的概率。有 3×10000 个选择，（这里假设词汇表有10000个单词），最后再通过beam width大小选择前3个概率最大的输出对；



- Step 3~Step T：与Step2的过程是相似的，直到遇到句尾符号结束。



4. 集束搜索的改进

长度归一化：

对于集束搜索算法，我们的目标就是最大化下面的概率：

$$\arg\max_y \prod_{t=1}^{T_y} P(y^{<t>}|x, y^{<1>}, \ldots, y^{<t-1>}) = \arg\max_y P(y^{<1>}, \cdots, y^{<T_y>}|x)$$
$$= \arg\max_y P(y^{<1>}|x)P(y^{<2>}|x, y^{<1>}) \cdots P(y^{<T_y>}|x, y^{<1>}, \cdots, y^{<T_y-1>})$$

上面的得到的每一项一般都是很小的概率值，大量很小的概率值进行相乘，最后会得到更小的值，可能会造成数值下溢。所以在实践中，我们不会最大化上面这个公式的乘积，而是取log值，变成log求和最大值，得到一个数值上更加稳定的算法，公式如下：

$$\arg\max_y \sum_{y=1}^{T_y} \log P(y^{<t>}|x, y^{<1>}, \ldots, y^{<t-1>})$$

另外，我们还可以通过对上面的目标进行归一化，使其得到更好的效果。相比直接除以输出单词长度的值，可以使用更加柔和的方式：在 T_y 上加上一个指数 α，如 $\alpha = 0.7$，通过调整其大小获得更加好的效果。如果这里不使用长度归一化，模型会倾向于输出更短的翻译句子，对结果产生影响。

$$\frac{1}{T_y^{\alpha}}\arg\max_y \sum_{y=1}^{T_y} \log P(y^{<t>}|x, y^{<1>}, \ldots, y^{<t-1>})$$

通过上面的目标，选取得分最大的句子，即为我们的模型最后得到的输出结果。

集束搜索讨论：

- Beam width：B的选择，B越大考虑的情况越多，但是所需要进行的计算量也就相应的越大。在常见的产品系统中，一般设置B = 10，而更大的值（如100，1000，...）则需要对应用的领域和场景进行选择。
- 相比于算法范畴中的搜索算法像BFS或者DFS这些精确的搜索算法，Beam Search 算法运行的速度很快，但是不能保证找到目标准确的最大值。

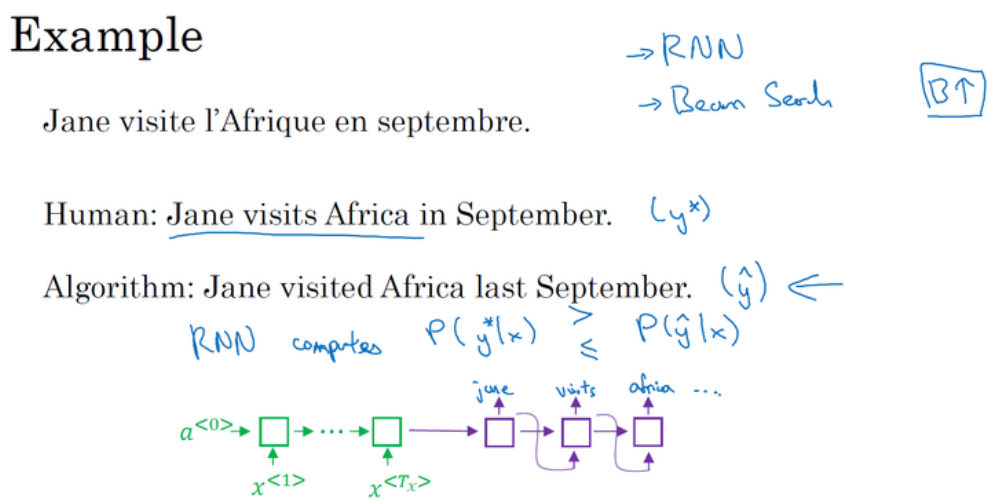
5. 集束搜索的误差分析

集束搜索算法是一种近似搜索算法，也被称为启发式搜索算法。它的输出不能保证总是可能性最大的句子，因为其每一步中仅记录着Beam width为3或者10或者100种的可能的句子。

所以，如果我们的集束搜素算法出现错误了要怎么办呢？我们如何确定是算法出现了错误还是模型出现了错误呢？此时集束搜索算法的误差分析就显示出了作用。

例子：

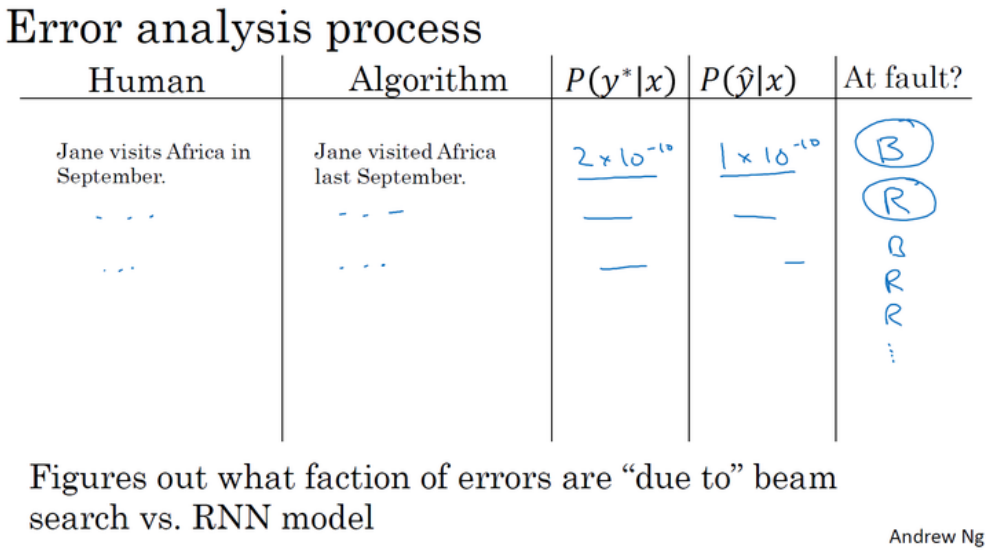
同样以法语句子的英文翻译为例子，我们人类对法语句子的翻译如中间的句子，而我们的模型输出的翻译如下面的句子。通过我们的模型，我们分别计算人类翻译的概率 $P(y^*|x)$ 以及模型翻译的概率 $P(\hat{y}|x)$，比较两个概率的大小，通过比较我们就能知道是因为Beam Search 算法的问题还是RNN模型的问题。如下图所示：



- $P(y^*|x) > P(\hat{y}|x)$ 的情况：Beam search算法选择了 $\hat{y}$，但是 y^* 却得到了更高的概率，所以Beam search 算法出错了；
- $P(y^*|x) <= P(\hat{y}|x)$ 的情况：翻译结果 y^* 相比 $\hat{y}$ 要更好，但是RNN模型却预测 $P(y^*|x) < P(\hat{y}|x)$，所以这里是RNN模型出现了错误。

过程：

在开发集上，对各个句子进行检测，得到每个句子对应的出错情况，那么根据整个开发集的上算法错误和模型错误的比例，就可以针对性地对二者之一进行改进和修正了。

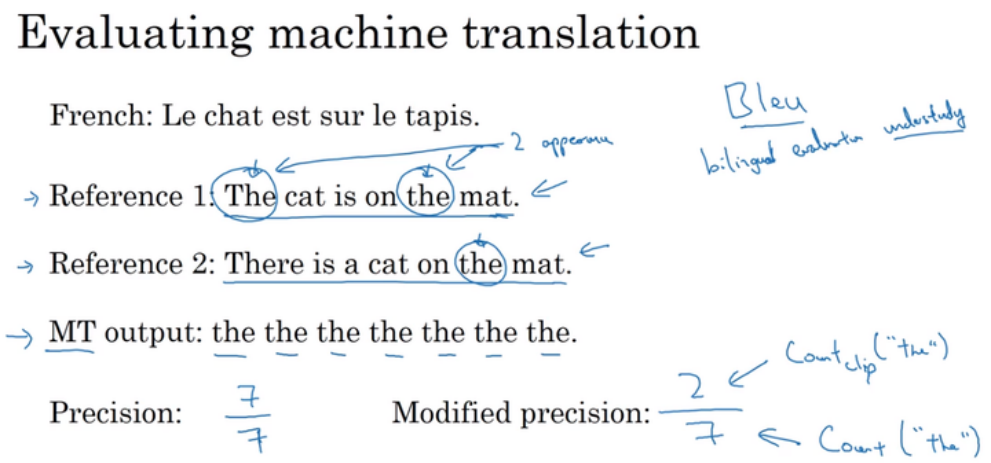


6. Bleu Score

对于机器翻译系统来说，一种语言对于另外一种语言的翻译常常有多种正确且合适的翻译，我们无法做到像图像识别一样有固定准确度答案，所以针对不同的翻译结果，往往很难评估那一个结果是更好的，哪一个翻译系统是更加有效的。这里引入Bleu score 用来评估翻译系统的准确性。（Bleu, bilingual evaluation understudy）

评估机器翻译：

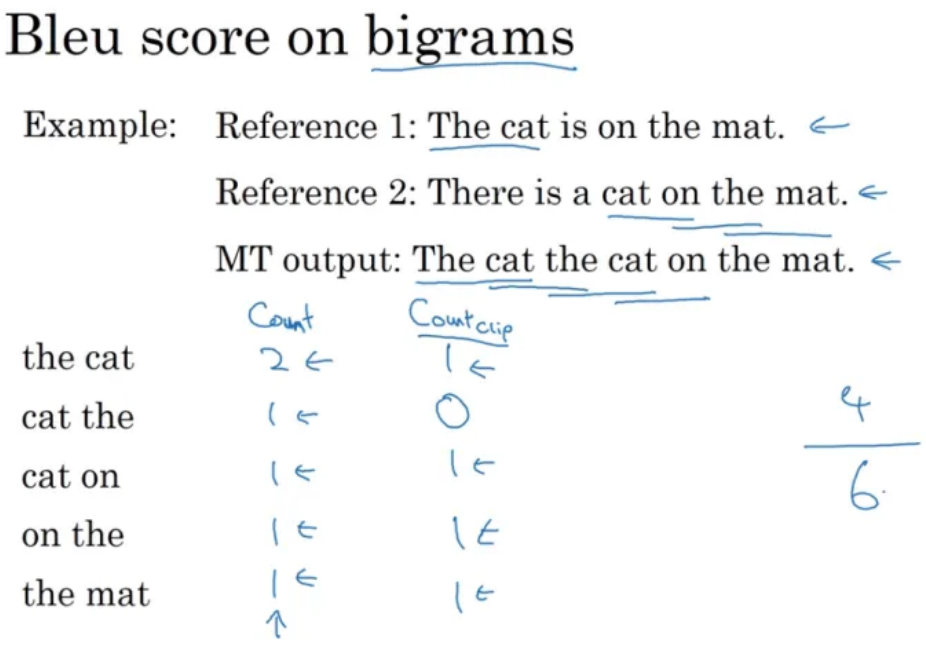
如下面的法语翻译的例子，我们有两种不同的翻译，但是两种翻译都是正确且较好的翻译结果。Bleu score 的评估理念是观察机器生成的翻译结果中的每一个词是否出现在至少一个人工翻译结果的参考之中。（这些参考位于开发集或者测试集中）。



- 精确度：观察输出结果的每一个词是否出现在参考中。但是对于图中机器翻译的糟糕的结果，精确度确非常高。
- 改良的精确度：将每个单词设置一个得分上限（单个参考句子中出现的最大的次数，如图中的the单词的上限为2）。

二元词组的Bleu score：

与单个词的评估相似，这里我们以两个相邻的单词作为一个二元词组来进行Bleu得分评估，得到机器翻译的二元词组的得分和其相应的得分上限，进而得到改进的精确度。



对于不同的n-gram，我们计算改良的精确度得分的公式如下：

$$P_1=\frac{\sum_{unigram\in\hat{y}} Count_{clip}(unigram)}{\sum_{unigram\in\hat{y}} Count(unigram)}$$

$$P_n=\frac{\sum_{n-gram\in\hat{y}} Count_{clip}(n-gram)}{\sum_{n-gram\in\hat{y}} Count(n-gram)}$$

Bleu score细节：

- 得到每种n-gram的Bleu score： P_n ，一般有 P_1，P_2，P_3，P_4；
- 组合Bleu scores：

$$BP\exp(\frac{1}{4}\sum_{n=1}^{4}P_n)$$

其中，BP(brevity penalty)简短惩罚，作为一个调节因子，来对太短的翻译结果的翻译系统进行惩罚。

$$BP=\begin{cases}1, if\ MT\_length > reference\_length\\ \exp(1-MT\_length/reference\_length), otherwise\end{cases}$$

Bleu score 作为机器翻译系统的一种单一评估指标，它有一个虽然不是非常完美，但是却也非常好的效果，其加快了整个机器翻译领域的进程，对机器翻译具有革命性的影响。同时，Bleu score对大多数的文本生成的模型均是有效的评估手段。

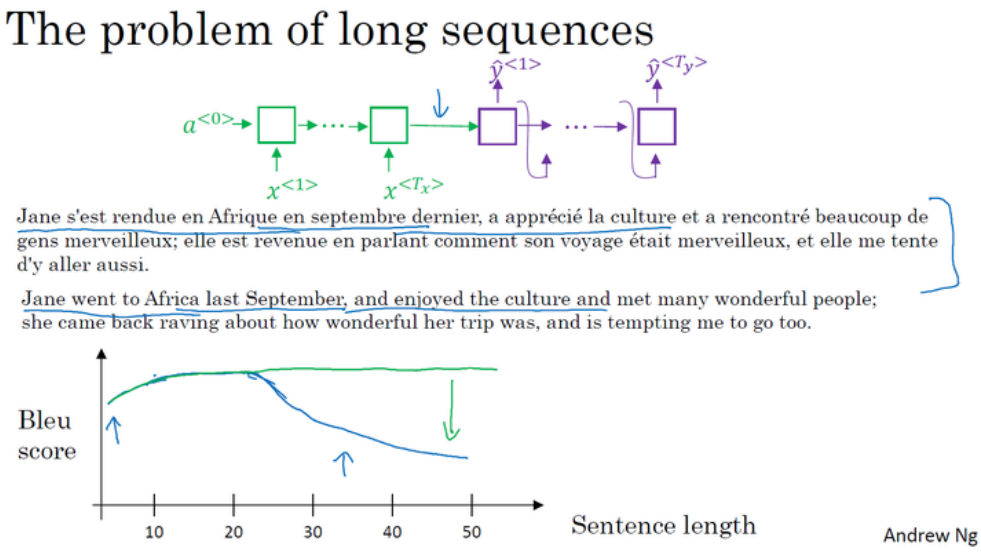
7. 注意力模型

长句子存在的问题：

利用我们前面的编码和解码的RNN模型，我们能够实现较为准确度机器翻译结果。对于短句子来说，其性能是十分良好的，但是如果是很长的句子，翻译的结果就会变差。

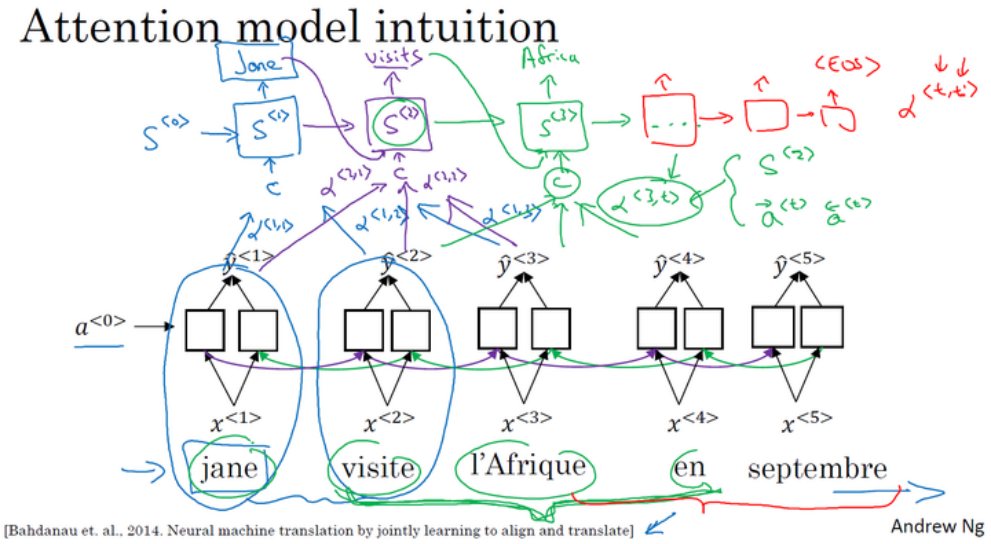
对于我们人类进行人工翻译的时候，我们所做的也不是像编码解码RNN模型一样记忆整个输入句子，再进行相应的输出，因为记忆整个长句子是很难的，所以我们是一部分一部分地进行翻译。编

码解码RNN模型的结构，其Bleu score会在句子长度超过一定值的时候就会下降，如图中的蓝色线所示。而引入的注意力机制，和人类的翻译过程非常相似，其也是一部分一部分地进行长句子的翻译，而其得到的翻译结果的Bleu曲线则如图中绿色线所示：



注意力模型的简介：

对于法语翻译英语的例子中，针对每个单词的输出，一般来说与某个输出相关的或者有较大影响的单词应该是集中在附近的几个单词或者说是某些部分。所以注意力模型会在输入的每个输入的信息块上计算注意力权重，不同的权重对每一步的输出结果有不同的注意力影响。



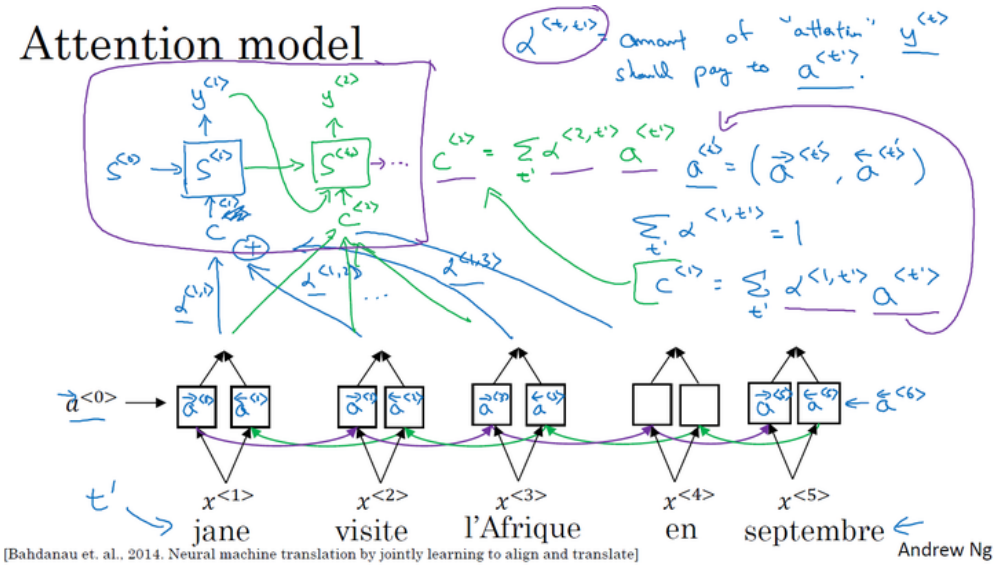
注意力模型的详细介绍：

这里我们以一个双向的RNN模型来对法语进行翻译，得到相应的英语句子。其中的每个RNN单元均是LSTM或者GRU单元。

对于双向RNN，通过前向和后向的传播，我们可以得到每个时间步的前向和后向的激活值 $\overrightarrow{a}^{<t'>}$，$\overleftarrow{a}^{<t'>}$，这里以 $a^{<t'>}=(\overrightarrow{a}^{<t'>},\overleftarrow{a}^{<t'>})$ 来表示输入句子通过双向RNN得到的每一步的激活值。

同时，对于英文输出，以另外一个RNN模型来构建，通过对输入单词的注意力权重 $\alpha^{<t,t'>}$，可以得到一个有关输入上下文的加权和 $C^{<t>}$。其中，注意力权重满足条件：$\sum_{t'}\alpha^{<t,t'>}=1$，注意力权重为对于每个输出 $y^{<t>}$ 需要在每个输入单词的激活值 $a^{<t'>}$ 上的关注度，得到相应的上

下文 $c^{<t>} = \sum_{t'} \alpha^{<t,t'>} a^{<t'>}$ 。到这里，我们可以发现对于注意力权重的设置就显得至关重要。

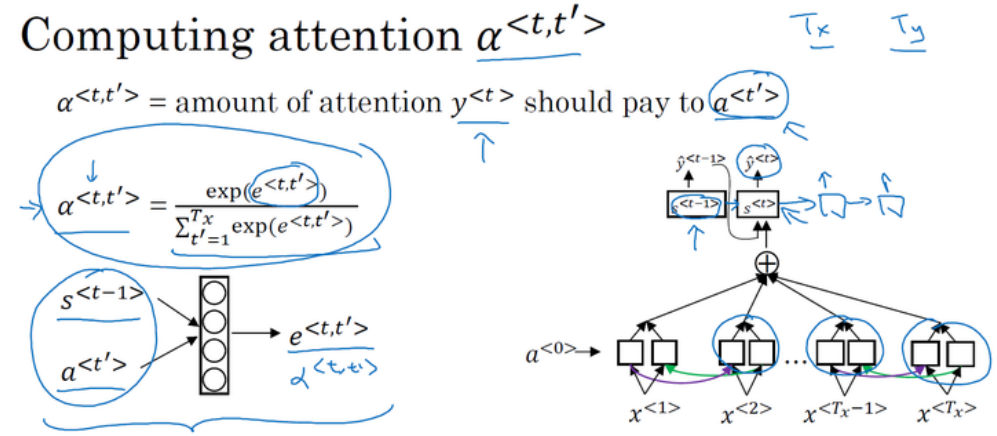


注意力权重的计算：

注意力权重： $\alpha^{<t,t'>}$ ，表示输出 $y^{<t>}$ 需要在每个输出激活值 $a^{<t'>}$ 上付出的注意力大小。计算注意力权重的公式如下，确保了权重和为1：

$$a^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t'=1}^{T_x} \exp(e^{<t,t'>})}$$

而其中的 $e^{<t,t'>}$ 则是通过一层神经网络来进行计算得到的，其值取决于输出RNN中前一步的激活值 $s^{<t-1>}$ 以及输入RNN当前步的激活值 $a^{<t'>}$ 。我们可以通过训练这个小的神经网络模型，使用反向传播算法来学习一个对应的关系函数，如下图所示。



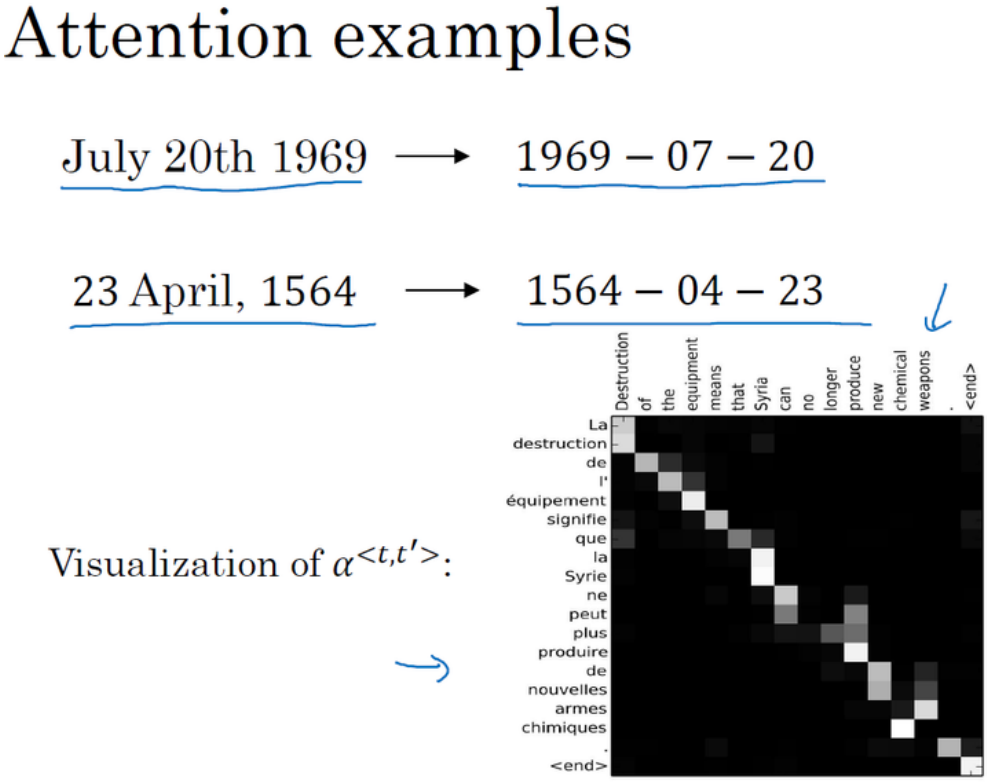
该模型的一个缺点就是时间复杂度较高，具有 $O(T_x T_y)$ 的复杂度。但是在机器翻译的应用中，输入和输出的句子一般不会太长，这样的复杂度是可以接受的。

相关论文：

- Bahdanau et. al., 2014. Neural machine translation by jointly learning to align and translate;
- Xu et. al., 2015. Show, attend and tell: Neural image caption generation with visual attention。

注意力的例子：

- 将不标准的时间格式转换为统一的时间格式；
- 对注意力权重进行可视化。



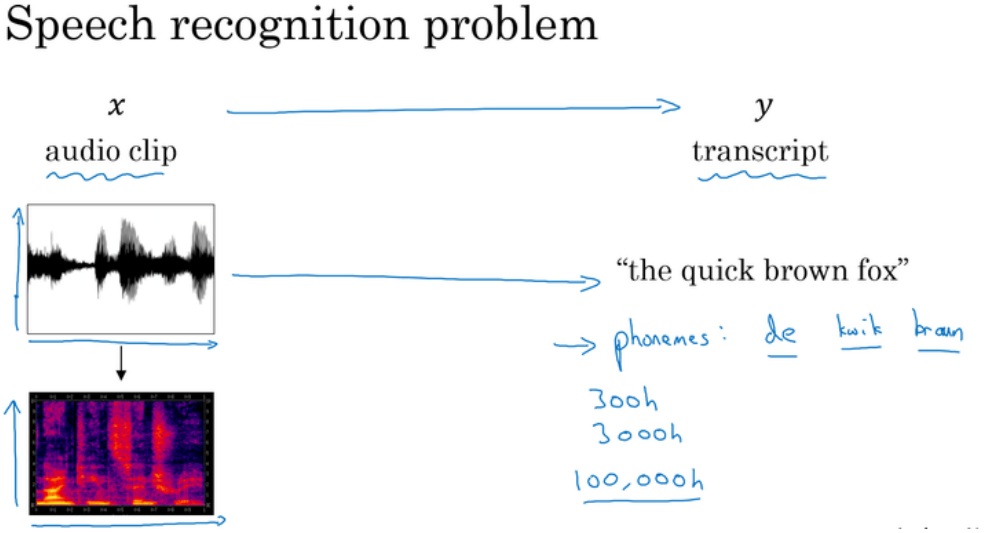
8. 语音识别

语音识别是将一段音频信号转化为相应的文本信息。

语音识别问题：

在语音识别问题上，借鉴于人耳的处理过程，会将音频信号转化为该音频信号的声谱图（时间和频率的图，不同颜色代表不同的频谱能量的大小），再将声谱图作为特征送到相应的算法中进行处理。

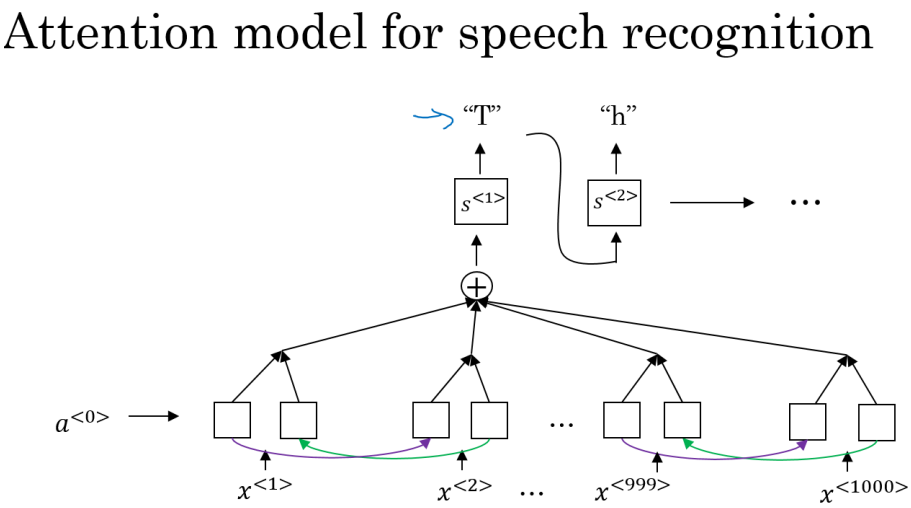
之前的语音识别的系统中，语言学家构造了音位来学习语音识别系统。但是随着深度学习的发展，在end-to-end的模型中，这种音位的表示法已经不重要了。我们完全可以做到直接从语音信号到文本信息的转化，无需人工特征的设计。



目前的语音识别的应用中，常见的语音数据大小为300h、3000h，而在一些大型的更加精确的语音系统中，已经应用了上万小时的语音数据。

注意力模型的语音识别：

为了构建一个语音识别的系统，我们可以应用前面介绍的注意力模型：

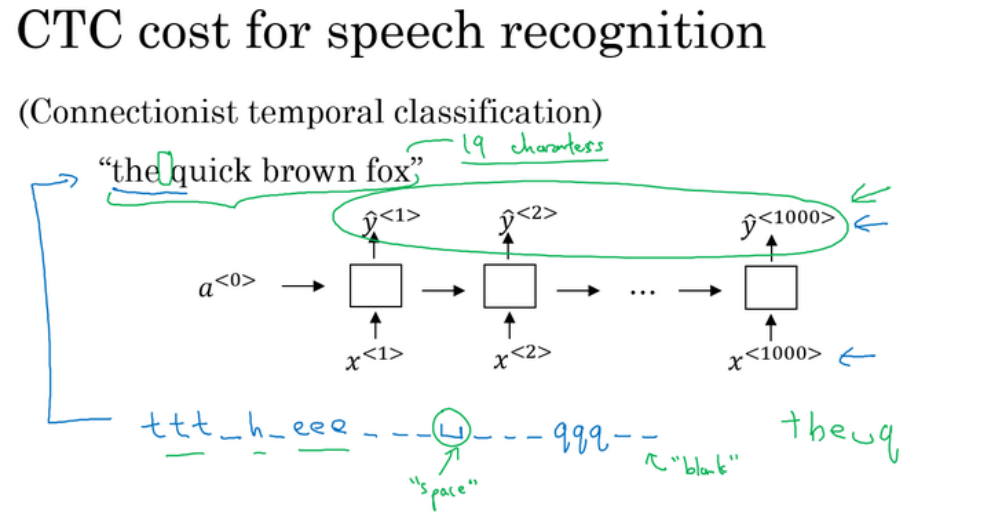


CTC 损失函数的语音识别：

同时，另外一种效果较好的就是使用CTC损失函数的语音识别模型。（CTC，Connectionist temporal classification）。

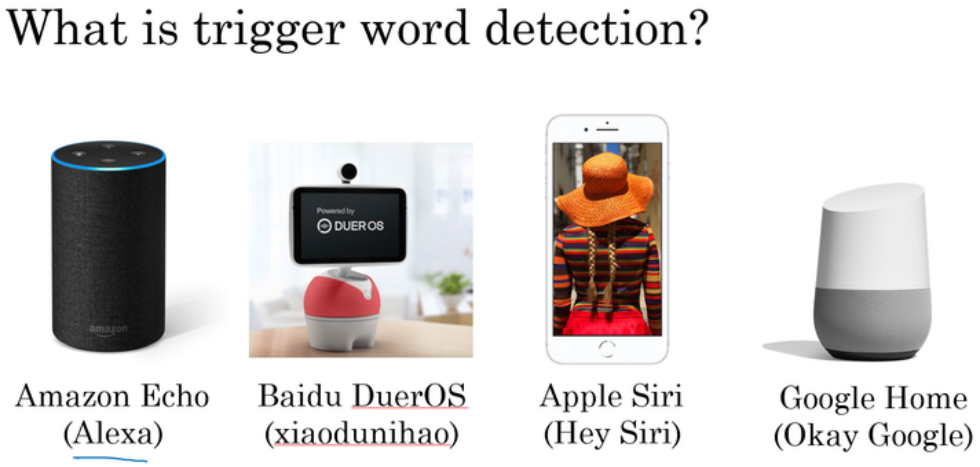
对于语音识别系统来说，我们的输入是音频信号，而输出是文本信息，对于音频信号，我们以小的时间间隔进行频率采样，可能对于一个10s的语音片段，我们就能够得到1000个特征的输入片段，而往往我们的输出仅仅是几个单词。

对于上面的问题，在CTC损失函数中，允许我们的RNN模型输出有重复的字符和插入空白符的方式，强制使得我们的输出和输入的大小保持一致，如下图所示。



9. 触发字检测

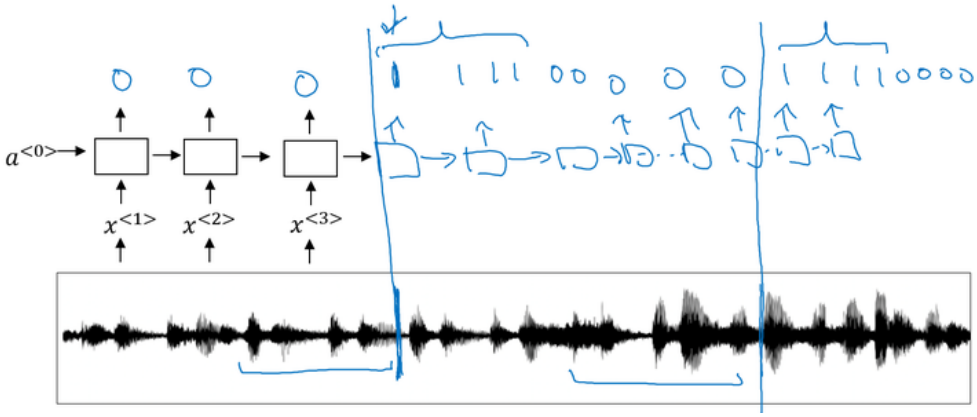
随着目前深度学习的发展，越来越多的智能设备需要通过一些关键语音来进行唤醒的操作，相应的这些系统可以被称作触发字检测系统。如下面的一些设备均属于触发字检测系统的应用：



触发字检测算法：

一种可以简单应用的触发字检测算法，就是使用RNN模型，将音频信号进行声谱图转化得到图像特征或者使用音频特征，输入到RNN中作为我们的输入。而输出的标签，我们可以以触发字前的输出都标记为0，触发字后的输出则标记为1。

Trigger word detection algorithm



上面方法的缺点就是0、1标签的不均衡。一种简单的方法就是在触发字后的多个目标标签都标记为1，在一定程度上可以提高系统的精确度。

编辑于 2018-04-12 15:24

人工智能　深度学习（Deep Learning）　RNN