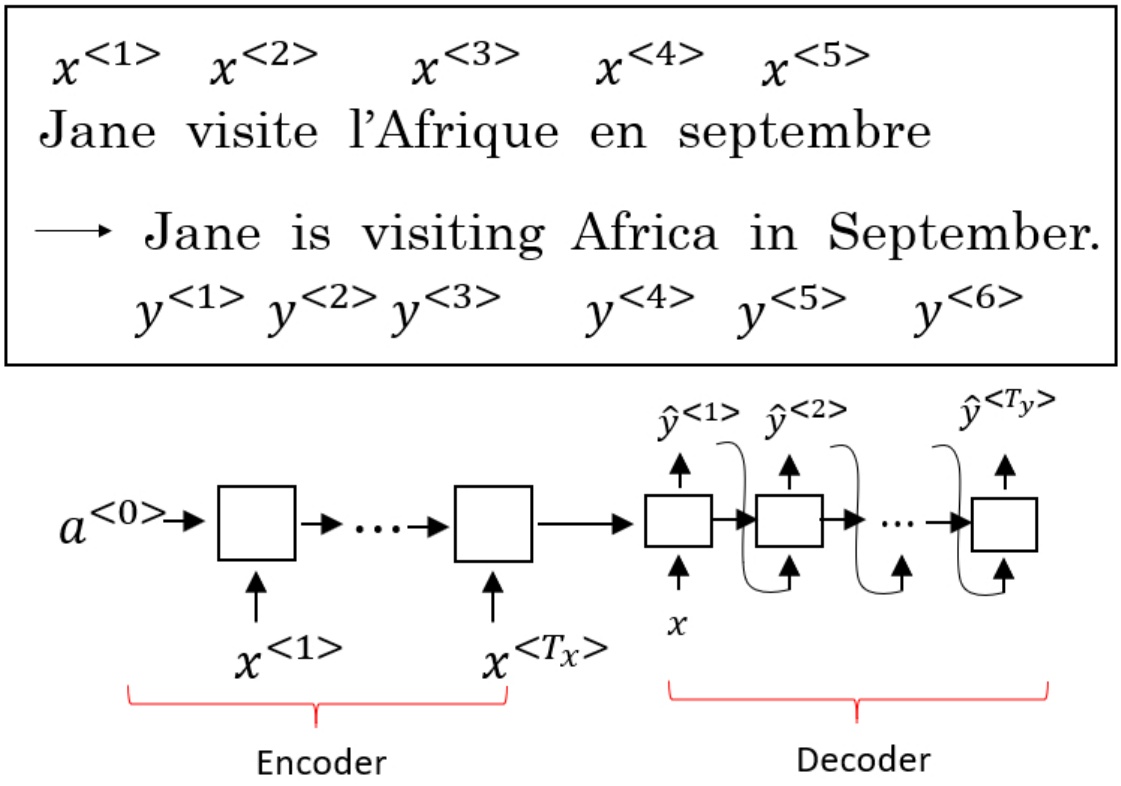
本周主要讲了Seq2Seq模型、注意力机制以及Seq2Seq的应用。

1. 基本模型

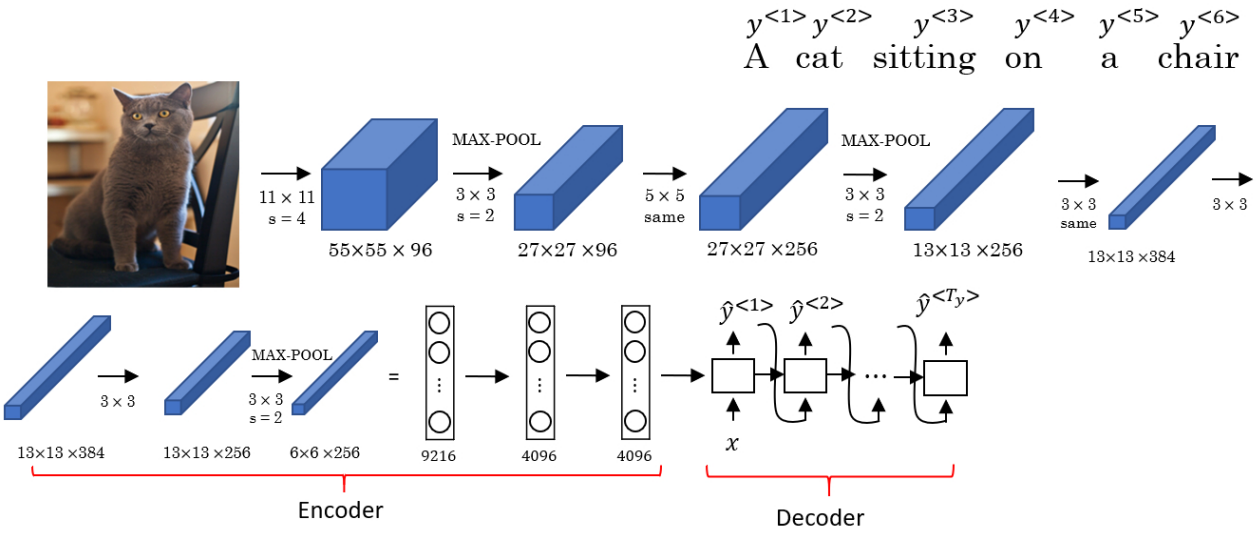
Seq2Seq（Sequence-to-Sequence）模型能够应用于机器翻译、语音识别等各种序列到序列的转换问题。一个Seq2Seq模型包含**编码器（Encoder）**和**解码器（Decoder）**两部分，它们通常是两个不同的RNN。如下图所示，将编码器的输出作为解码器的输入，由解码器负责输出正确的翻译结果。



参考论文：

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

这种编码器-解码器的结构也可以用于图像描述（Image captioning）。将 AlexNet 作为编码器，最后一层的 Softmax 换成一个 RNN 作为解码器，网络的输出序列就是对图像的一个描述。

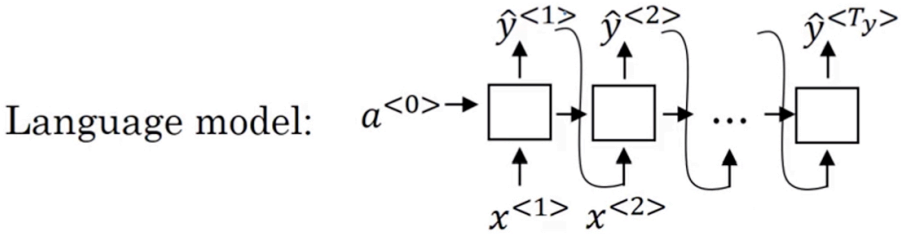


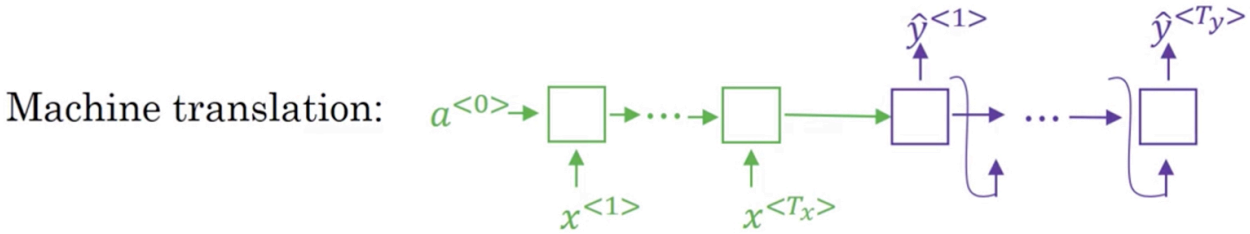
图像描述的相关论文：

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

2. 选择最有可能的句子

机器翻译用到的模型与语言模型相似，只是用编码器的输出作为解码器第一个时间步的输入（而非0）。因此机器翻译的过程其实相当于建立一个条件语言模型。





$$p(y^{<1>},\cdots,y^{<T_y>}|x^{<1>},\cdots,x^{<T_x>})$$

由于解码器进行随机采样，输出的翻译结果可能有好有坏。因此需要找条件概率最大的翻译：

$$\arg\max_{y^{<1>}\ldots y^{<T_y>}} P\left(y^{\langle 1\rangle},\ldots,y^{\langle T_y\rangle}\mid x\right)$$

不使用贪心搜索的原因：

- 贪心搜索不能保证整个翻译语句是最优的；
- 单词库有成百上千万的词汇，计算每一中单词的组合是不可行的。

更常用的是使用**集束搜索（Beam Search）**算法。

3. Beam Search

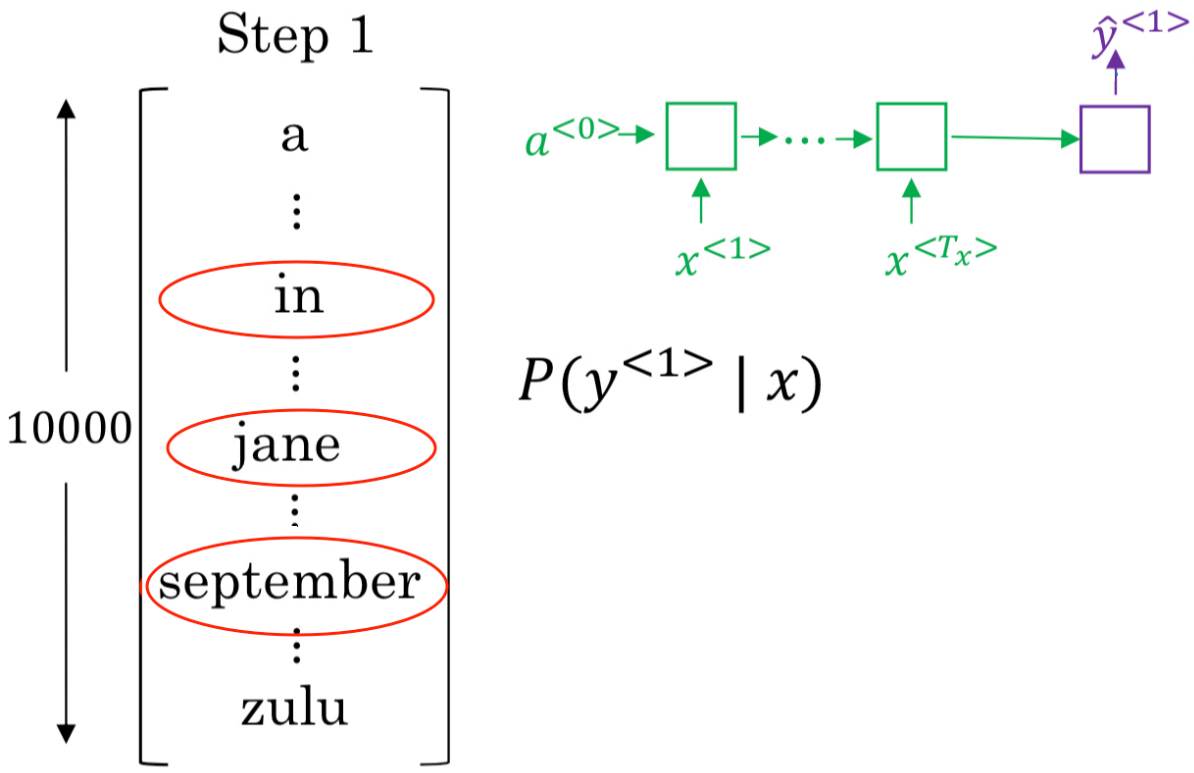
集束搜索（Beam Search）会在每个时间步考虑多个可能的选择。设定一个**集束宽带(Beam width)B**，代表了在每个时间步选择最优的B个单词，并进行缓存，以待后续用。

还是以翻译下面的法语为例，对Beam Search算法的过程进行说明，设$\text{Beam width} = 3$：

> 法语：Jane visite l'Afrique en septembre.

- **Step1**

  将法语句子输入Encoder网络中，得到句子的编码。将句子编码输入到Decoder网络中，在第一个时间步，可以得到词汇表中每个单词的输出概率。选取前3个单词，然后保存起来：



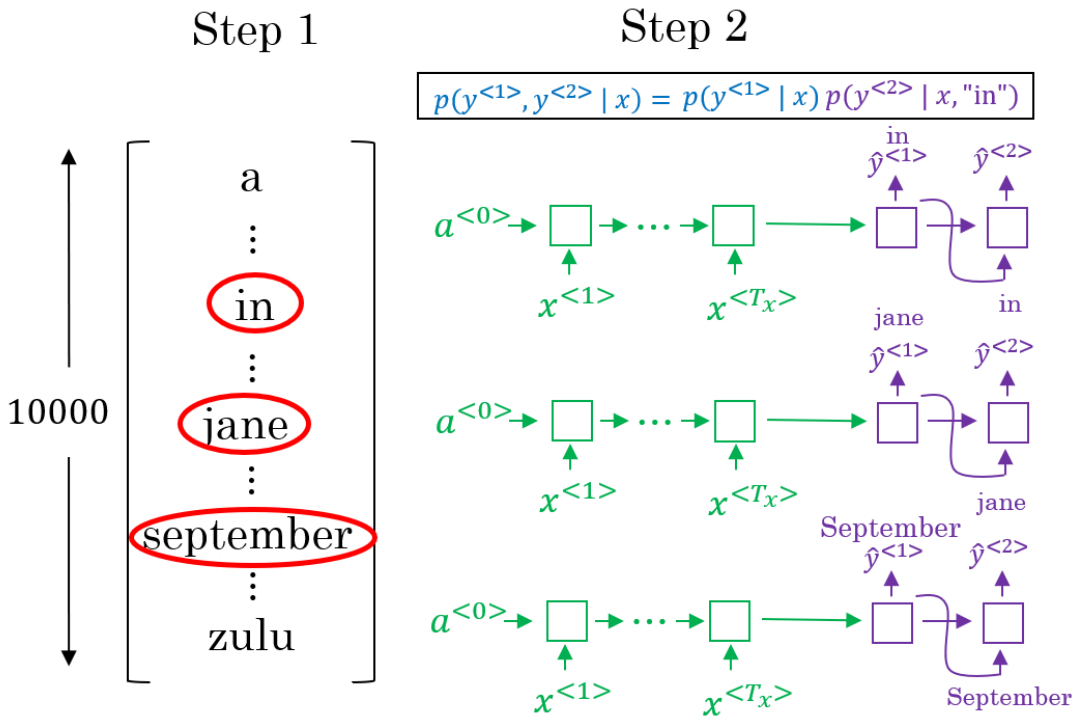
- **Step2**

  将第一步得到的3个预选词作为第二个时间步的输入，得到$p\left(\hat{y}^{<2>}|x,\hat{y}^{<1>}\right)$

但我们需要的是第一个词和第二个词的联合概率最大，可根据条件概率计算得到：

$$P\left(\hat{y}^{\langle 1\rangle}, \hat{y}^{\langle 2\rangle} \mid x\right)=P\left(\hat{y}^{\langle 1\rangle} \mid x\right) P\left(\hat{y}^{\langle 2\rangle} \mid x, \hat{y}^{\langle 1\rangle}\right)$$

设词典中有10000个单词，则当$B=3$时，有3×10000个$p\left(\hat{y}^{<2>} | x, \hat{y}^{<1>}\right)$，仍然取概率最大的前3个词对。



- **Step3 ~ StepT:**

  与Step2的计算类似，直到遇到句尾符号$\langle \mathrm{EOS} \rangle$ 结束。

可以看到，当$B=1$时，即为贪心搜索。

4. 改进Beam Search

4.1 长度归一化

对于集束搜索算法，我们的优化目标是：

$$\begin{aligned}\arg\max_{y} \prod_{t=1}^{T_y} P\left(y^{<t>} \mid x, y^{<1>}, \ldots, y^{<t-1>}\right) &= \arg\max_{y} P\left(y^{<1>}, \cdots, y^{<T_y>} \mid x\right) \\ &= \arg\max_{y} P\left(y^{<1>} \mid x\right) P\left(y^{<2>} \mid x, y^{<1>}\right) \\ &\ldots P\left(y^{<T_y>} \mid x, y^{<1>}, \ldots, y^{<T_y-1>}\right)\end{aligned}$$

上面的每一项都是很小的数，多个很小的数相乘，可能导致数值下溢（Numerical Underflow），对等式两边取log，得到数值计算上更稳定的算法：

$$\arg\max_{y} \sum_{t=1}^{T_y} \log P\left(y^{<t>} \mid x, y^{<1>}, \ldots, y^{<t-1>}\right)$$

但上面的优化目标有个问题：倾向于输出更短的翻译，使用长度归一化，减小对输出长结果的惩罚：

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

T_y输出的单词数，α超参数，如取$\alpha = 0.7$

4.2 Beam Width

Beam Width越大，考虑的情况就越多，所需的计算量也越大。一般$B = 10$，更大的B需结合实际的领域和场景。

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. Beam Search误差分析

因为Beam Search是一种启发式搜索，每一步仅记录Beam Width个记录，因此不能保证最终结果是最优的。当结合 Seq2Seq 模型和集束搜索算法所构建的系统出错（没有输出最佳翻译结果）时，我们通过误差分析来分析错误出现在 RNN 模型还是集束搜索算法中。

例如，对于下述有人工和算法得到的两个翻译：

$$\text{Human: Jane visits Africa in September.}(y^*)$$
$$\text{Algorithm: Jane visited Africa last September.}(\hat{y})$$

- $p(y^* \mid x) > p(\hat{y} \mid x)$：$y^*$出现的概率更高，但Beam Search却选择了$\hat{y}$，说明Beam Search算法错误；
- $p(y^* \mid x) < p(\hat{y} \mid x)$：$y^*$的结果本应比$\hat{y}$更好，但RNN却预测$\hat{y}$更好，因此RNN算法错误。

建立一个如下图所示的表格，记录对每一个错误的分析，有助于判断错误出现在 RNN 模型还是集束搜索算法中。如果错误出现在集束搜索算法中，可以考虑增大集束宽 BB；否则，需要进一步分析，看是需要正则化、更多数据或是尝试一个不同的网络结构。

Human	Algorithm	$P(y^*\|x)$	$P(\hat{y}\|x)$	At fault?
Jane visits Africa in September.	Jane visited Africa last September.	2×10^{-10}	1×10^{-10}	B
				R
				B
				R
				R
				⋮

6. Bleu得分

Bleu（Bilingual Evaluation Understudy）得分用于评估机器翻译的质量，其思想是机器翻译的结果越接近于人工翻译，则评分越高。

French: Le chat est sur le tapis.

Reference 1: The cat is on the mat.

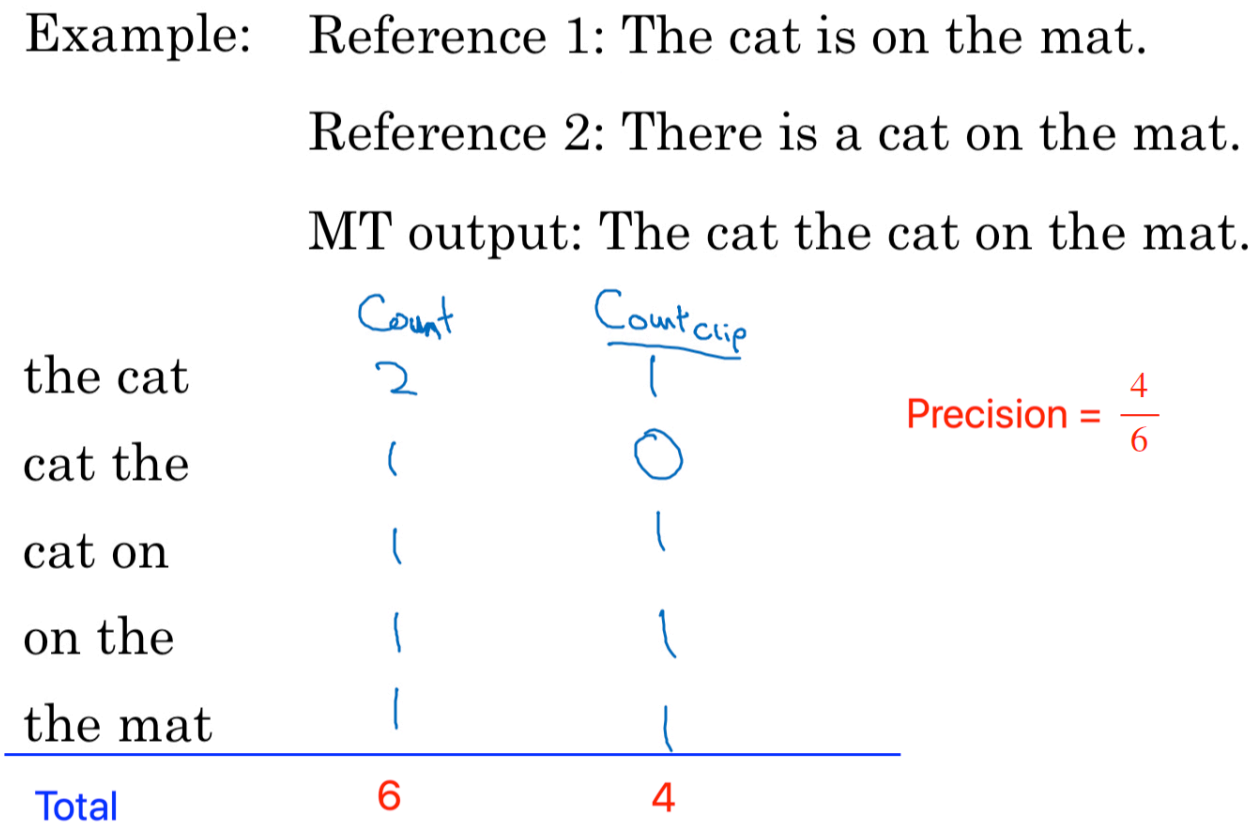
Reference 2: There is a cat on the mat.

MT output: the the the the the the the.

Precision: $\frac{7}{7}$ Modified precision: $\frac{2}{7}$ $\frac{\text{count}_{\text{clip}}(\text{"the"})}{\text{count}(\text{"the"})}$

- Precision：观察输出的单词出现在Reference中的比例，对于图中这样糟糕的MT output，Precision却等于1，不合理！
- Modify precision：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）。

上述方法是以单个单词为单位进行统计，以单个单词为单位的集合称为**unigram**；而以成对单词为单位的集合称为**bigram**. 还上面的例子，对于二元组，计算其Bleu Score：



对于不同的$n-gram$，其改进的Bleu Score计算公式如下：

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

组合几个Bleu Score得到最终的Bleu Score：

$$\text{Bleu} = \text{BP}\cdot\exp\left(\frac{1}{N}\sum_{n=1}^{N}\log P_n\right)$$

其中，BP是简短惩罚(Brevity Penalty, BP):

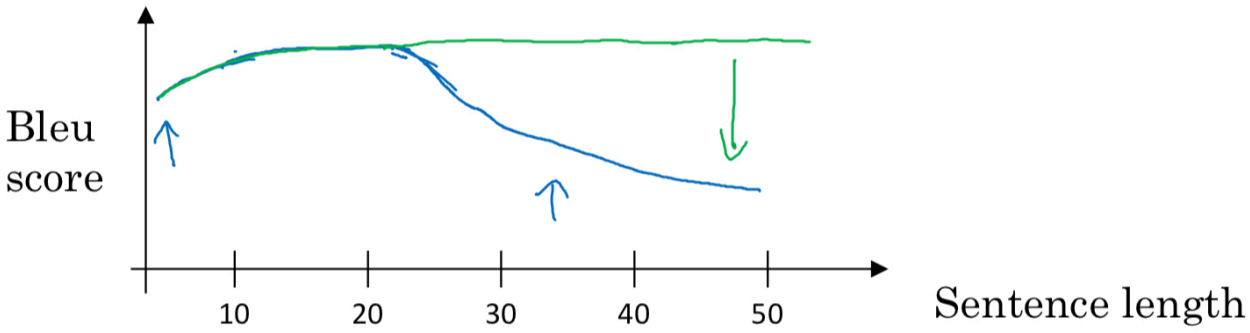
$$BP = \begin{cases} 1, & MT\_\text{length} \geq \text{Reference\_length} \\ \exp\left(1-\frac{MT\_\text{length}}{\text{Reference\_length}}\right), & MT\_\text{length} < \text{Reference\_length} \end{cases}$$

Bleu score 作为机器翻译系统的一种**单一实数评估指标**，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. Attention Model

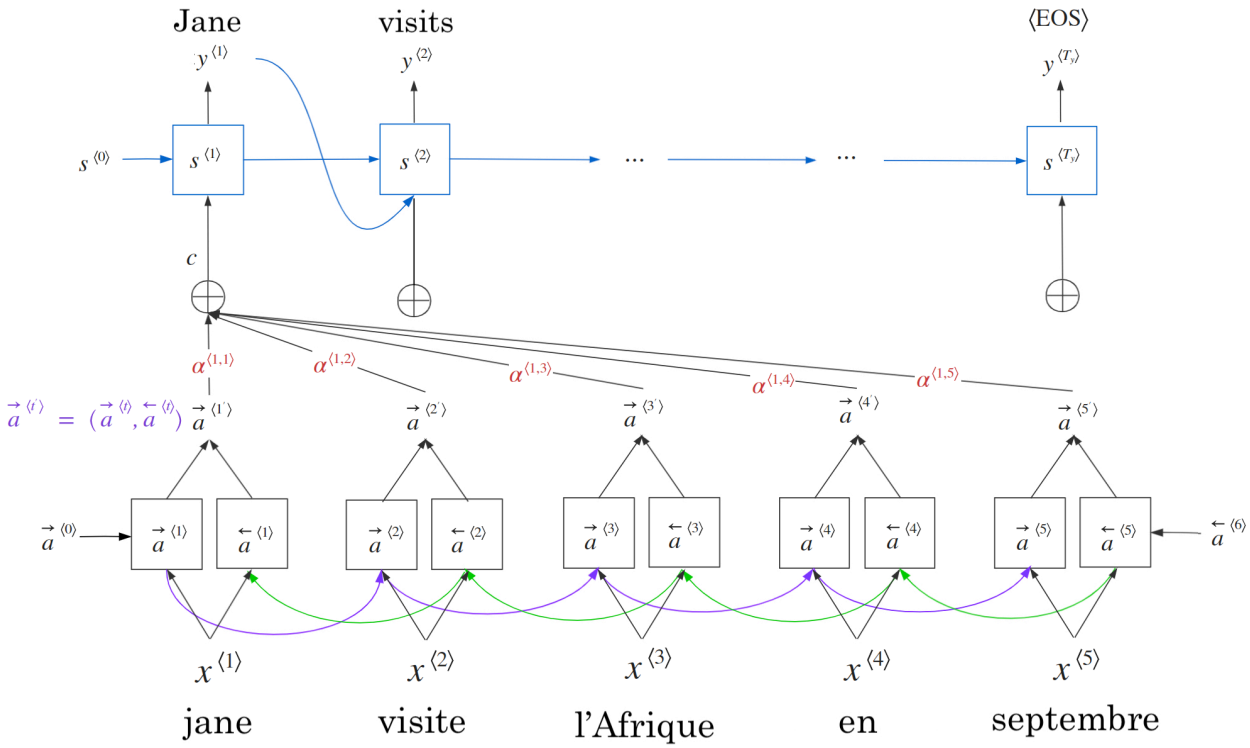
7.1 长句翻译存在的问题

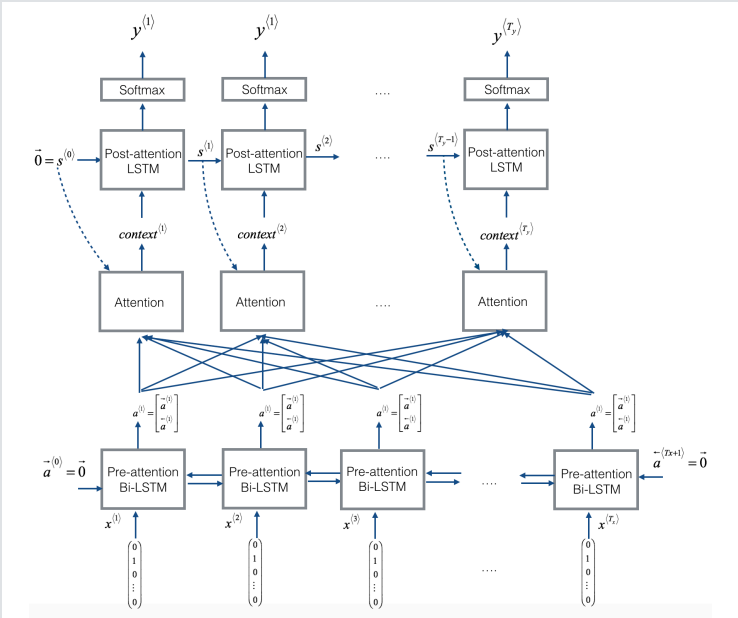
前面讲到的Encoder-Decoder的Seq2Seq模型，对于较短语句的翻译，表现不错，但随着句子越长，Bleu Score呈下降趋势：

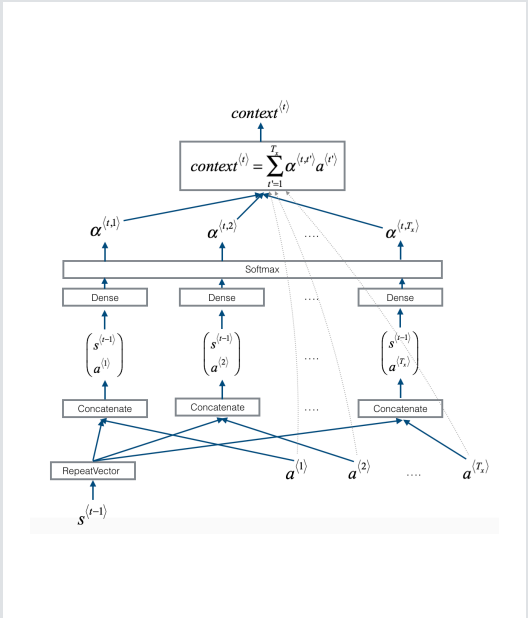


对于我们人类进行人工翻译的时候，我们所做的也不是像编码解码RNN模型一样记忆整个输入句子，再进行相应的输出，因为记忆整个长句子是很难的，所以我们是一部分一部分地进行翻译。编码解码RNN模型的结构，其Bleu score会在句子长度超过一定值的时候就会下降，如图中的蓝色线所示。而引入的注意力机制，和人类的翻译过程非常相似，其也是一部分一部分地进行长句子的翻译，而其得到的翻译结果的Bleu曲线则如图中绿色线所示。

7.2 注意力模型概览







注意力模型的一个网络示意图如上图所示。底层是一个双向RNN，每个RNN单元可以是LSTM也可以使GRU。每一个时间步的激活包括前向和反向传播产生的激活：

$$a^{\langle t' \rangle} = \left(\overrightarrow{a}^{\langle t' \rangle}, \overleftarrow{a}^{\langle t' \rangle} \right)$$

顶层是一个“many to many”的RNN，第t 时间步的输入有上一时间步的激活值$s^{\langle t-1 \rangle}$ 、输出$y^{\langle t-1 \rangle}$ 以及底层BRNN的激活c：

$$c^{\langle t \rangle} = \sum_{t'} \alpha^{\langle t,t' \rangle} a^{\langle t' \rangle}$$

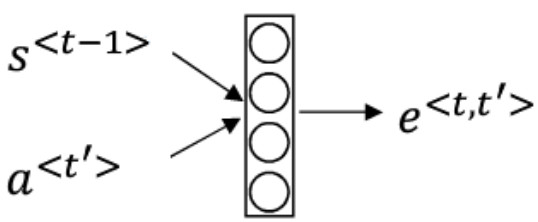
其中，$\alpha^{\langle t,t' \rangle}$表示$y^{\langle t \rangle}$对$a^{\langle t' \rangle}$的注意力：

$$\sum_{t'} \alpha^{\langle t,t' \rangle} = 1$$

我们使用Softmax确保上式成立：

$$\alpha^{\langle t,t' \rangle} = \frac{\exp\left(e^{\langle t,t' \rangle}\right)}{\sum_{t'=1}^{T_x} \exp\left(e^{\langle t,t' \rangle}\right)}$$

而$e^{\langle t,t' \rangle}$用一个神经网络学得，如下图：



注意力模型的一个缺点是时间复杂度为$O(n^3)$。

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：将注意力模型应用到图像标注中

8. Speech Recognition

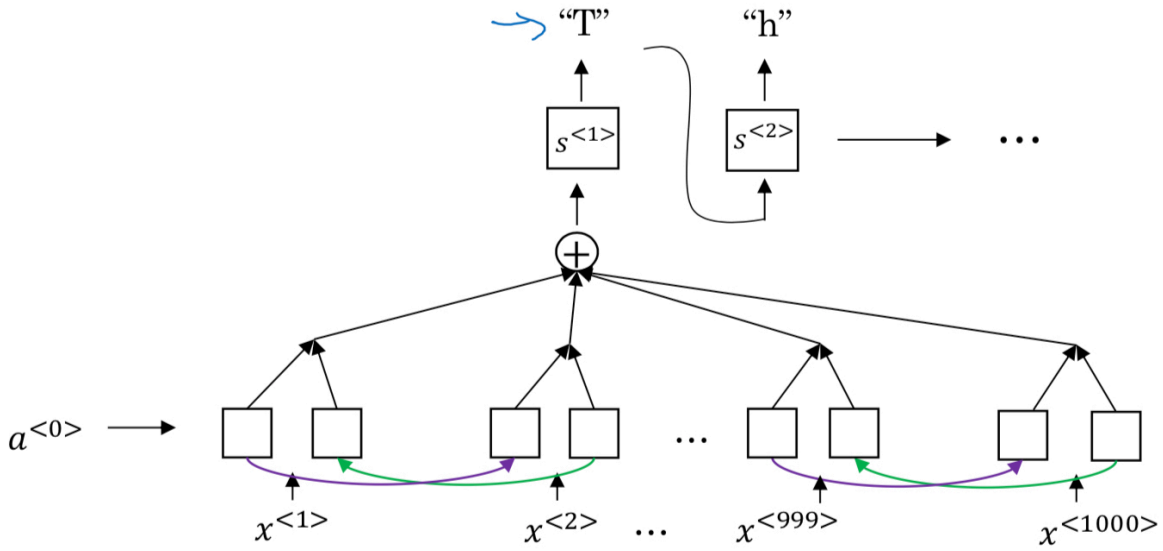
在语音识别任务中，输入是一段以时间为横轴的音频片段，输出是文本。

音频数据的常见预处理步骤是运行音频片段来生成一个**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），并将其作为特征。以前的语音识别系统通过语言学家人工设计的**音素（Phonemes）**来构建，音素指的是一种语言中能区别两个词的最小语音单位。现在的端到端系统中，用深度学习就可以实现输入音频，直接输出文本。

对于训练基于深度学习的语音识别系统，大规模的数据集是必要的。学术研究中通常使用 3000 小时长度的音频数据，而商业应用则需要超过一万小时的数据。

注意力模型的语音识别系统：

语音识别系统可以用注意力模型来构建，一个简单的图例如下：

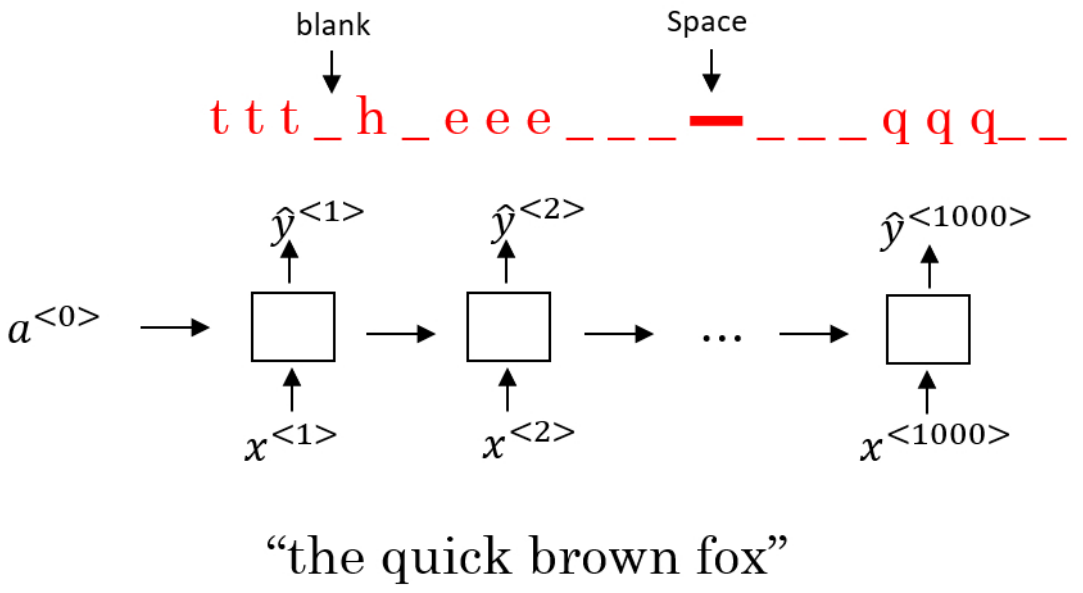


CTC 损失函数的语音识别：

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:



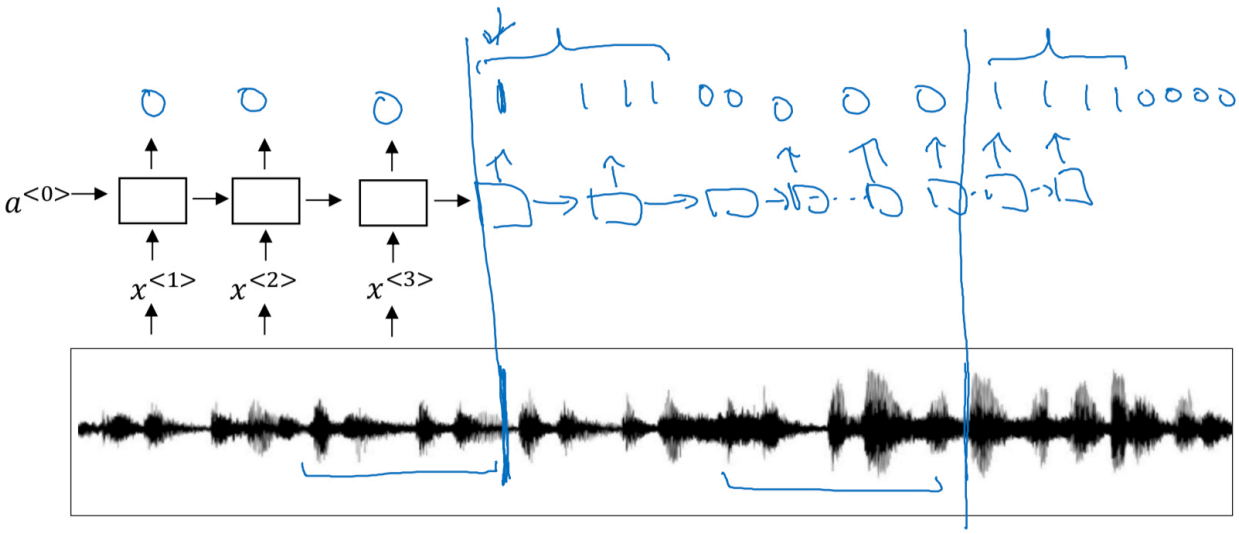
相关论文:

- Graves et al., 2006. Connectionist Temporal Classification: Labeling unsegmented sequence data with recurrent neural networks

9. 触发词检测

触发词检测（Trigger Word Detection）常用于各种智能设备，通过约定的触发词可以语音唤醒设备。

使用 RNN 来实现触发词检测时，可以将触发词对应的序列的标签设置为“1”，而将其他的标签设置为“0”。



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。