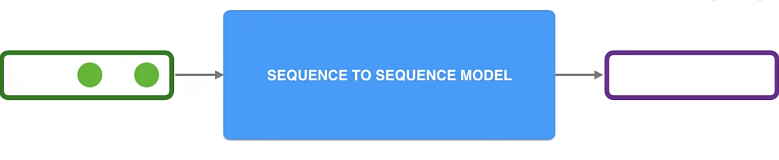
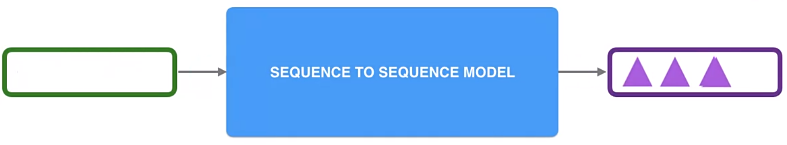
## Seq2seq与attention图解

### 1.seq2seq简介

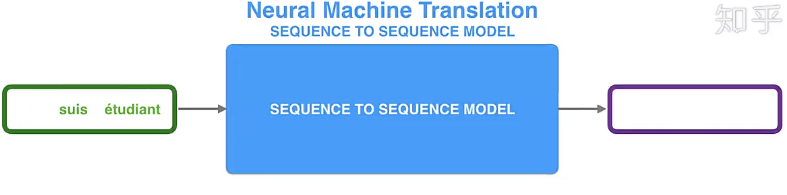
Seq2seq模型现在已经在机器翻译、文本摘要和图像解释方面取得了很大的成功。谷歌翻译在2016年使用这个模型。Seq2seq模型的输入是序列化数据（比如单词、信件内容、图片特征等），输出也是序列化数据。从模型与翻译，两个角度来叙述。（因doc文档难插入视频，很推荐这个作者的文档加视频，很简洁易懂）

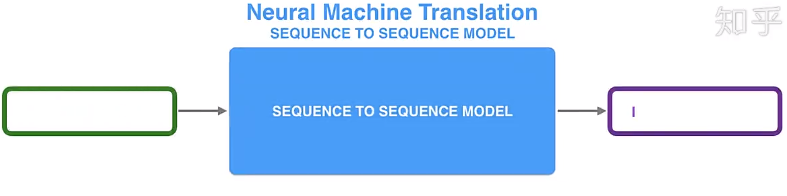
模型训练过程如下所示：

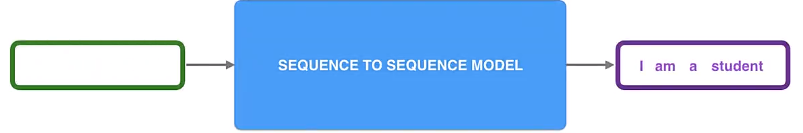




在机器翻译的网络中，输入序列是一串单词，一个接着一个进行输入，输出也是一系列单词,过程如下：



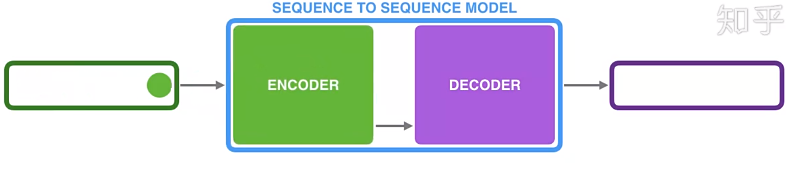


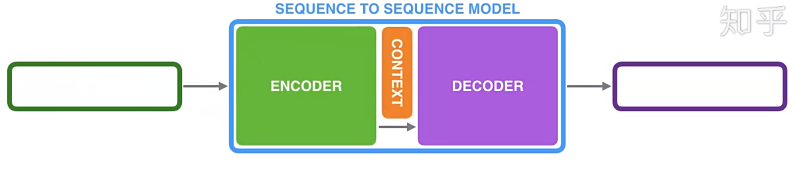


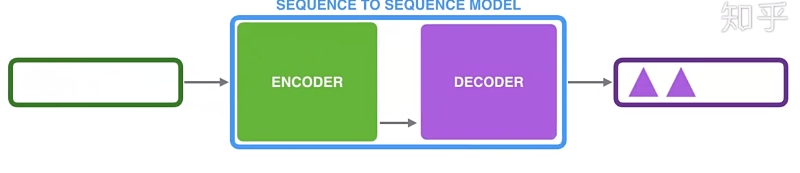
### 2.揭秘Seq2seq内部

Seq2seq是由 encoder和decoder两个部分组成。

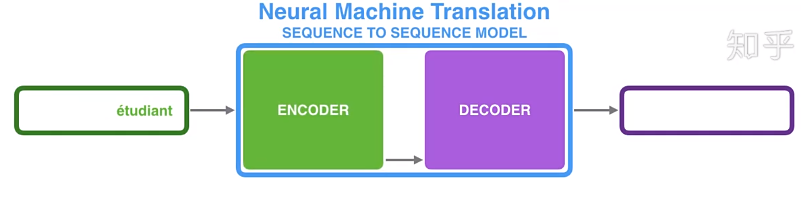
encoder对输入的内容进行处理，然后输出成一个向量，称为context, 在经过encoder之后生成的contenxt会被送到decoder部分，decoder会接收context然后产生输出序列。整体过程如下所示。

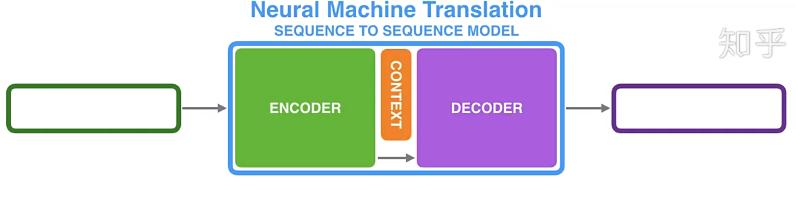


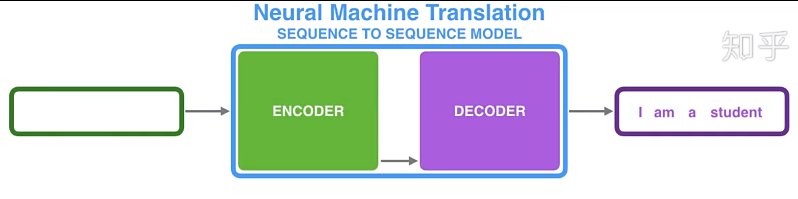




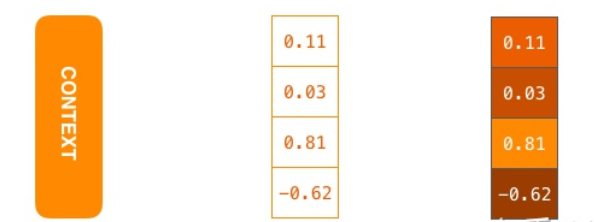
在机器翻译中，过程也是和上面一样，示例如下：





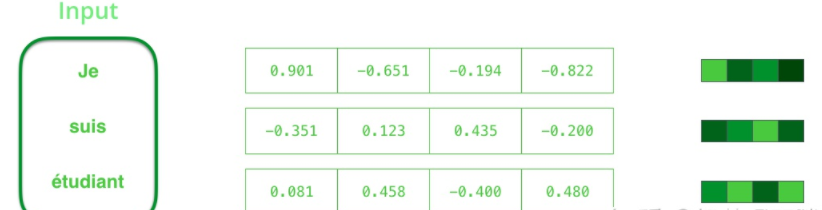


在机器翻译中，context是一个向量，其中的encoder和decoder一般都是Rnn网络.

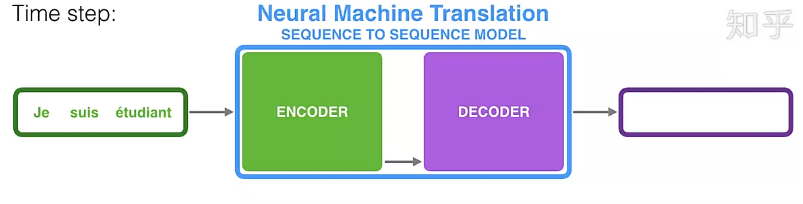


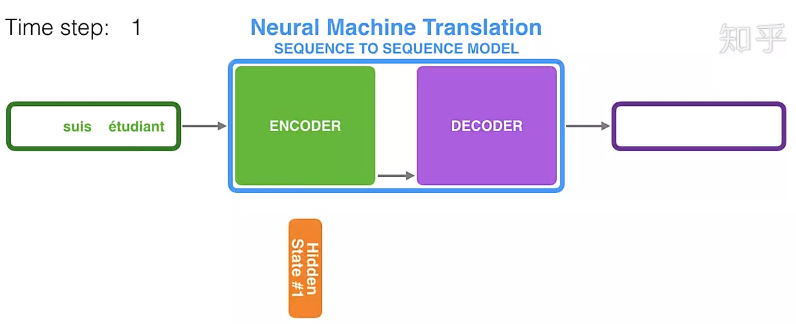
在设置模型时，可以设context的大小。它基本上是编码器RNN中隐藏单元的数量。这些可视化显示的向量大小为4，但在实际应用中，context的大小为256、512或1024.

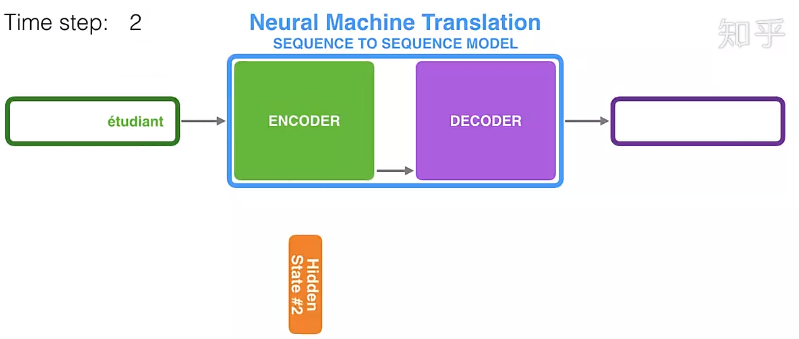
RNN的输入结构，每次输入需要输入一个隐藏单元和一个输入，在decoder中一个输入是一个单词，但是这个单词需要转化为向量。一般为了把单词转为向量，我们使用的一个方法叫做“word embedding”. 一般如下所示：

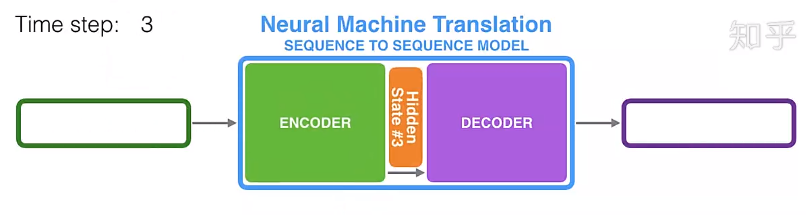


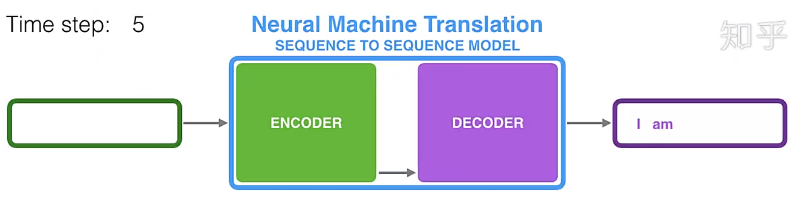
因为encoder和decoder都是RNN结构的网络，每一步RNN都做相同的处理，会根据之前的隐藏状态和输入来更新现在的隐藏状态，在下面的组图中,encoder输出的最后一个隐藏层就是我们之前提到的context.组图如下所示：

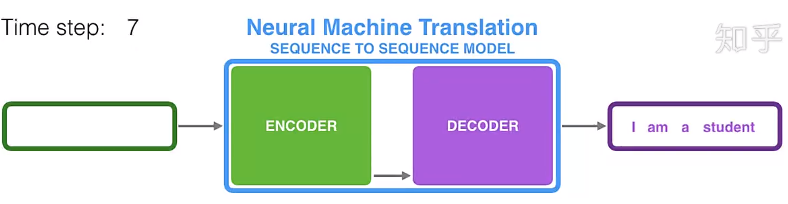




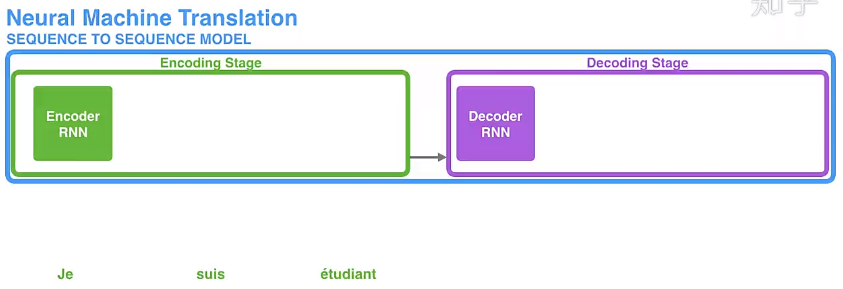


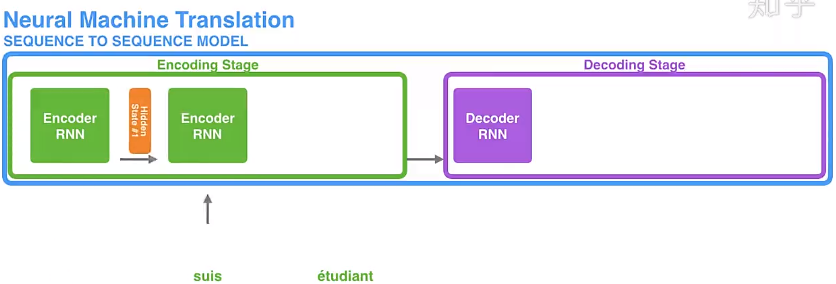


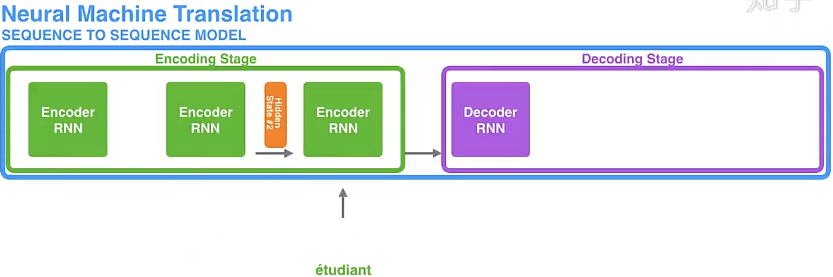


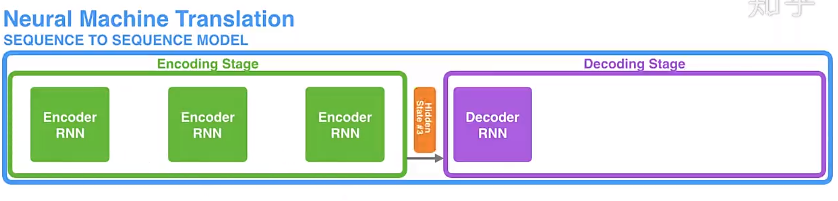


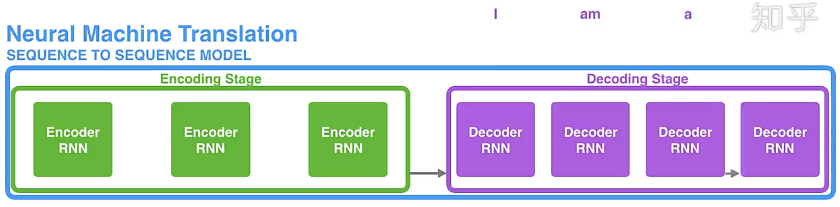
我们上面的组图和介绍，从大体上看只包含一个encoder和decoder,我们展开过程可以看的更加详细，组图如下：









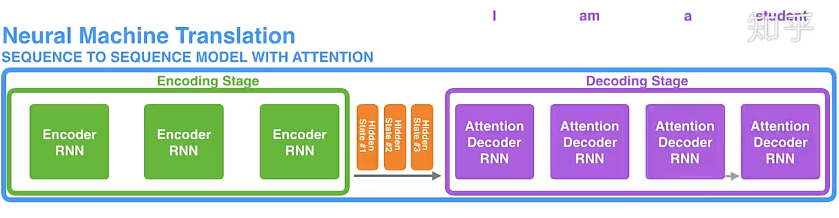


### 3.引入Attention

上面的模型使用的还依然是RNN，所以在处理长文本的时候比较困难，效果也不太理想，这个时候就引入了注意力机制，可以让模型只注意输入相关的部分。

加入注意力机制后，和以前的Seq2seq有两点不同。

**第一点：** 以前的Seq2seq是只传递最后一个的隐藏层，而加入Attention之后会传递所有隐藏层

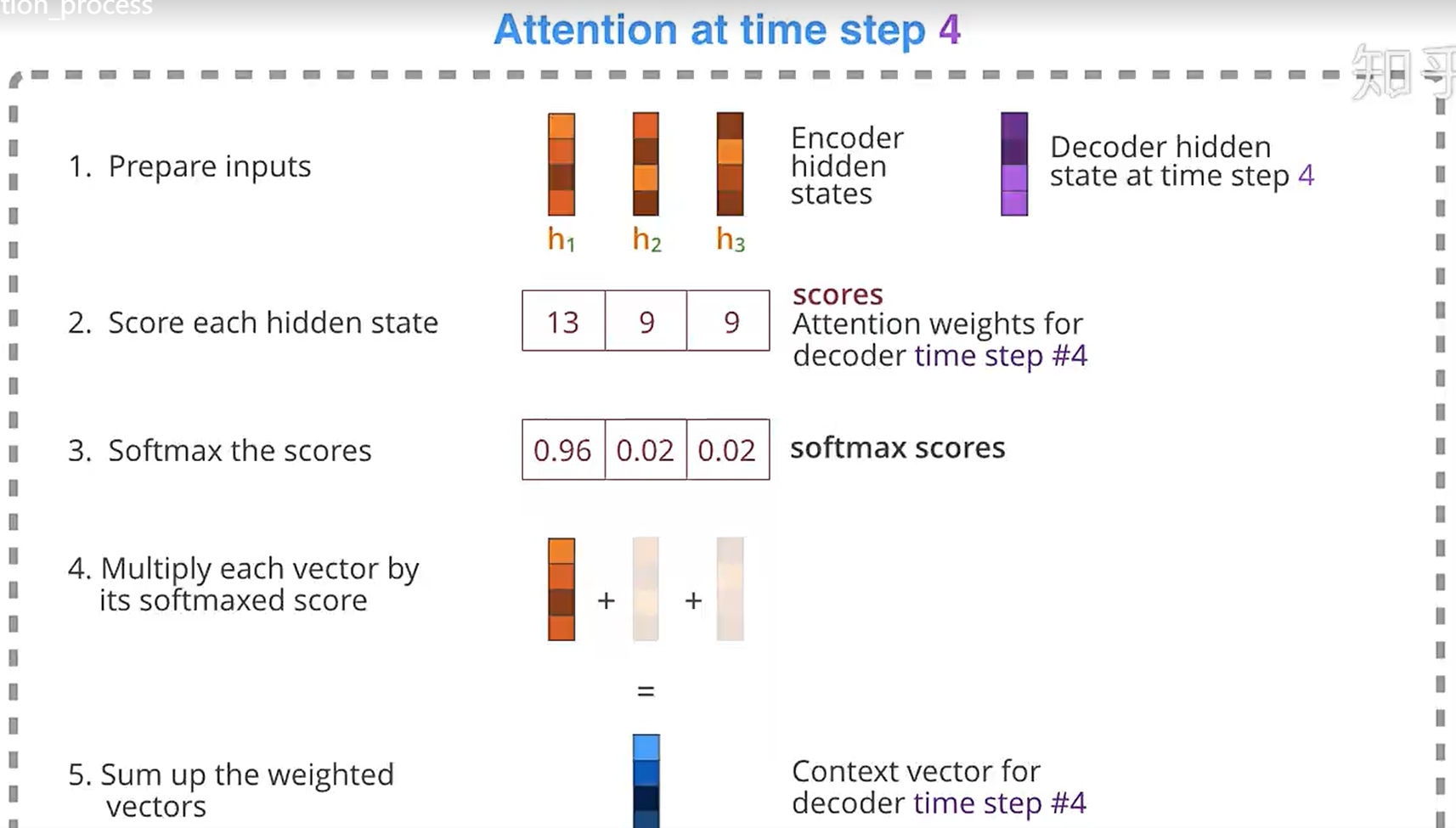


**第二点：**在decoder层，会有一些改变，主要有以下几点：

查看它接收到的一组encoder隐藏状态-每个encoder隐藏状态与输入语句中的某个单词最为相关

给每个隐藏层打分

用每个隐藏层状态去乘上得分，来凸显一些高分状态，隐藏低分状态。



接下来介绍输出过程：

RNN decoder 接收<END>和初始隐藏状态

经过RNN生成一个隐藏状态（h4），输出值被丢弃

用生成的h4经过Attention处理生成一个向量（C4)

拼接C4和h4得到新向量

把生成的新向量传递进一个前反馈神经网络

这一步的前反馈神经网络会生成一个单词

接下来重复这一步

