**# RNN\_LSTM\_GRU**

**1. 经典RNN**

* 在实际应用中，会遇到很多序列形的数据，例如

1. 自然语言处理问题: 可以看做是第一个单词，可以看做是第二个单词，依次类推。
2. 语音处理: 此时，是每帧的声音信号。
3. 时间序列问题: 例如每天的股票价格等。

* 序列形的数据就不太好用原始的神经网络处理了。为了建模序列问题，RNN引入了隐状态（hidden state）的概念，可以对序列形的数据提取特征，接着再转换为输出。（可以将视为包含了time-step的信息）

1. 先从的计算开始看:

绿色的钟表

中度可信度描述已自动生成

1. 的计算和类似。要注意的是，在计算时，每一步使用的参数都是一样的，也就是说每个步骤的参数都是共享的，这是**RNN的重要特点**。
2. 以下计算过程可以无限地持续下去。

钟表的特写

描述已自动生成

1. 得到输出值的方式就是直接通过进行计算

图示, 示意图

描述已自动生成

1. 完整结构

图示

描述已自动生成

* **输入和输出序列必须要是等长的，即N - N。**

由于这个限制的存在，经典RNN的适用范围比较小，但也有一些问题适合用经典的RNN结构建模，如：

1. 计算视频中每一帧的分类标签。因为要对每一帧进行计算，因此输入和输出序列等长。
2. 输入为字符，输出为下一个字符的概率，即著名的Char RNN。

**2. RNN变种(其他类型)**

**2.1 N - 1**

* 要处理的问题输入是一个序列，输出是一个单独的值而不是序列

-> 只在最后一个上进行输出变换就可以了

图示

描述已自动生成

* 这种结构通常用来处理序列分类问题。

1. 输入一段文字判别它所属的类别。
2. 输入一个句子判断其情感倾向。
3. 输入一段视频并判断它的类别。

……

**2.2 1 - N**

* 输入不是序列而输出为序列的情况 -> 只在序列开始进行输入计算

图示

描述已自动生成

* 还有一种结构是把输入信息X作为每个阶段的输入

图示

描述已自动生成

* 这种1 VS N的结构可以处理的问题有

1. 从图像生成文字(image caption)，此时输入的X就是图像的特征，而输出的y序列就是一段句子。
2. 从类别生成语音或音乐等。

**2.3 N - M**

* **RNN最重要的一个变种：N - M。**

**这种结构又叫Encoder-Decoder模型，也可以称之为Seq2Seq模型。**

* 原始的RNN (N-N)要求序列等长，然而我们遇到的大部分问题序列都是不等长的，如机器翻译中，源语言和目标语言的句子往往并没有相同的长度。

1. 为此，Encoder-Decoder结构先将输入数据编码成一个上下文向量

得到c有多种方式，最简单的方法就是把Encoder的最后一个隐状态赋值给，还可以对最后的隐状态做一个变换得到c，也可以对所有的隐状态做变换。

图示

描述已自动生成

1. 拿到之后，就用另一个RNN网络对其进行解码，这部分RNN网络被称为Decoder。具体做法就是将当做之前的初始状态输入到Decoder中。

图片包含 游戏机, 钟表, 物体

描述已自动生成

还有一种做法是将c当做每一步的输入。

手机屏幕截图

低可信度描述已自动生成

* 由于这种Encoder-Decoder结构不限制输入和输出的序列长度，因此应用的范围非常广泛。

1. 机器翻译: Encoder-Decoder的最经典应用，事实上这一结构就是在机器翻译领域最先提出的。
2. 文本摘要: 输入是一段文本序列，输出是这段文本序列的摘要序列。
3. 阅读理解: 将输入的文章和问题分别编码，再对其进行解码得到问题的答案。
4. 语音识别: 输入是语音信号序列，输出是文字序列。

……