循环神经网络

1. 基于编码-解码的序列到序列架构

此处讨论如何训练RNN，使其将输入序列映射到**不一定等长的输出序列。**

经常将RNN的输入称为"**上下文**"，用C表示此上下文。C可能是一个丐萝输入序列$X = (x^{(1)}, \ldots, x^{(n_x)})$的向量或向量序列。

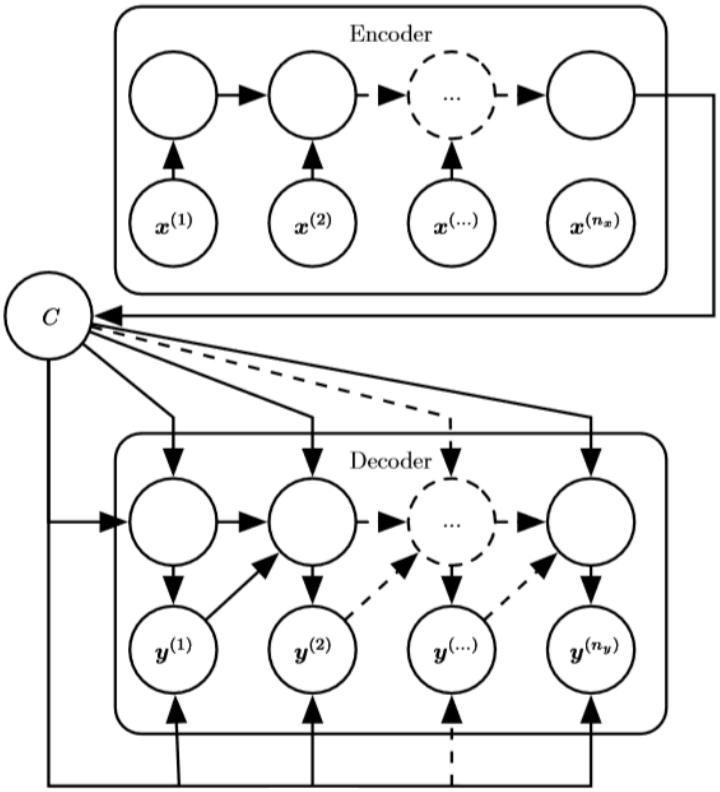

图 10.12: 在给定输入序列 $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n_x)})$ 的情况下学习生成输出序列 $(\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(n_y)})$ 的编码器-解码器或序列到序列的 RNN 架构的示例。它由读取输入序列的编码器 RNN 以及生成输出序列（或计算给定输出序列的概率）的解码器 RNN 组成。编码器 RNN 的最终隐藏状态用于计算一般为固定大小的上下文变量 C，C 表示输入序列的语义概要并且作为解码器 RNN 的输入。

该想法主要内容：

（1）**编码器**（encoder）或**读取器**（reader）或**输入**（input）RNN处理输入序列，编码器输出上下文C(通常是最终隐藏状态的简单函数)

（2）**解码器**（decoder）或**写入器**（writer）或**输出**（output）RNN则以固定长度的向量维为条件产生输出序列$\boldsymbol{Y} = (y^{(1)}, \ldots, y^{(n_y)})$。

此架构创新之处在于长度n_x和n_y可以彼此不同，而之前的架构约束$n_x = n_y = \tau$。

在序列到序列的架构中，两个RNN共同训练以最大化
$logP(y^{(1)}, \ldots, y^{(n_y)} | x^{(1)}, \ldots, x^{(n_x)})$（关于训练集中所有$x$, y对的平均）

向量到序列RNN至少有两种接受输入的方式。输入可以被提供为RNN的初始状态，或连接到每个时间步中的隐藏单元，且并不强制要求编码器与解码器是隐藏层具有相同的大小。

此架构的不足是：

编码器RNN输出的上下文C的维度太小而难以适当地概括一个长序列。为解决这个问题，可以通过一定方式将C称为可变长度的序列，而不是一个固定大小的向量，此外还可以引入将序列C的元素和输出序列的元素相关联的**注意力机制**(attention mechanism).