时间序列模型

> 序列模型是适用于序列数据（时序数据）的一类模型，常被用于语音识别以及自然语言处理领域，典型的算法如循环神经网络（RNN）。在语音识别领域，输入数据是一段音频，输出的是音频的文字部分，输入输出均为序列数据；**处理情感分类**问题时，输入的是一段评论性的文字，输出则是分类结果（如这段句子表示愤怒的情感）；**机器翻译问题**上，输入是一种语言的文本序列，输出是另一国语言的序列；**视频行为识别问题**中，输入是一系列的视频的帧，输出是对视频行为的判定结果；**命名实体识别问题**，输入一个句子，输出的是句子中的实体名称。上述的例子表明，序列模型有很多不同类型，有的序列模型输入输出都是序列，但有的模型只有输入或者输出是序列，因此，我们将分别对这些模型进行讨论。

1.序列模型应用

输入或者输出中包含有序列数据的模型叫做序列模型。以循环神经网络RNN为基础建立的序列模型在自然语言处理，语音识别等领域中引起了巨大的变革。以下是一些序列模型的典型应用：

语音识别： 输入输出都为序列。

音乐生成： 输出为序列。

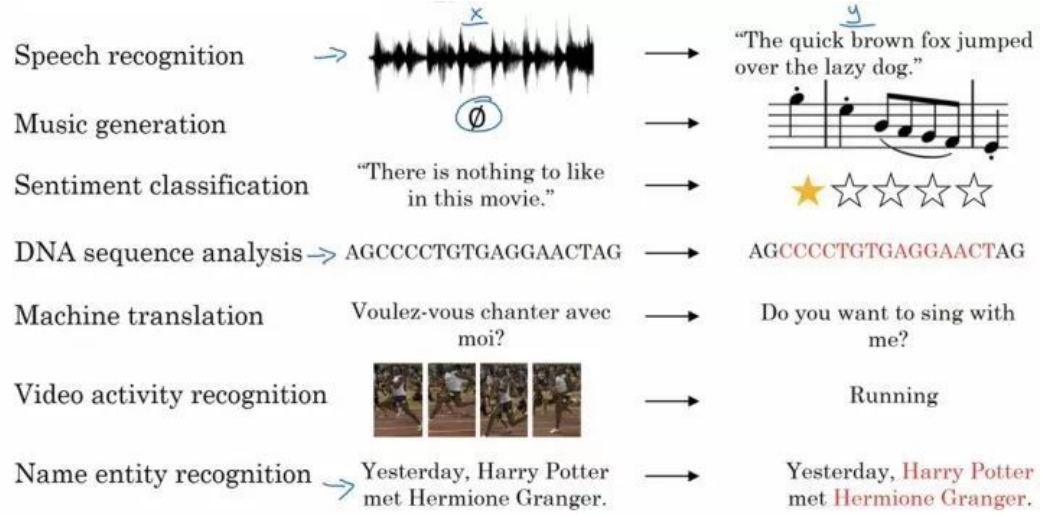
情感分析：输入为序列。

DNA序列分析：输入为序列。

机器翻译：输入输出都为序列。

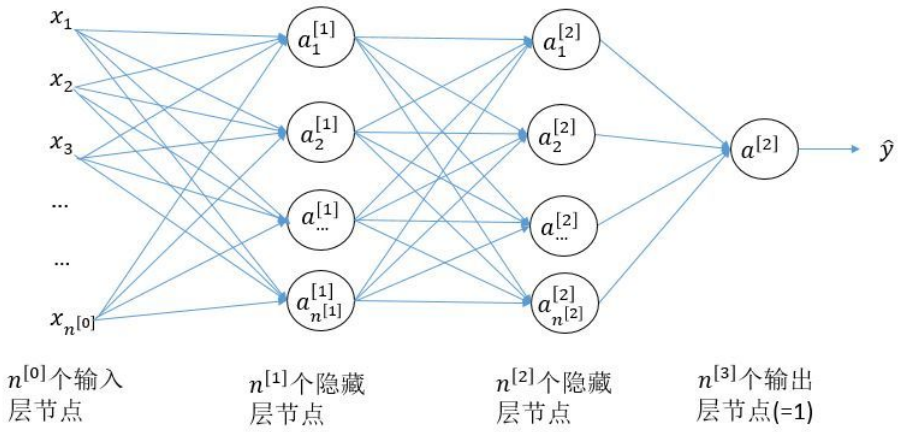
视频行为识别：输入为序列。

命名实体识别：输入输出都为序列。

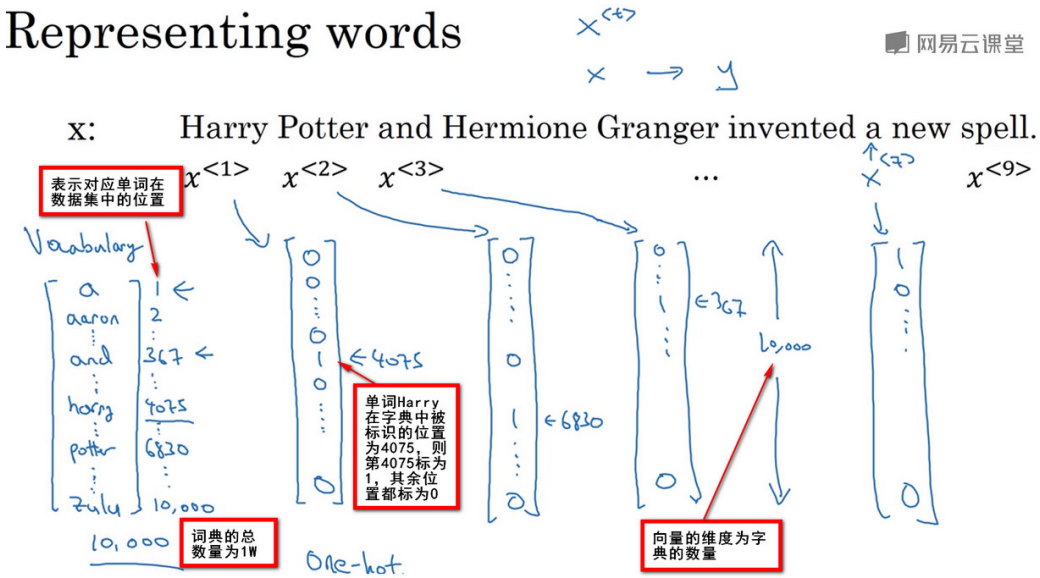


2.数学标记约定

在前面神经网络的数学书写中，我们遵守如下符号约定。不同节点node用下标来指定，不同样本sample用带小括号的上标来指定，不同层layer用带中括号的上标来指定，不同批次batch用带花括号的上标来指定。



在序列模型中，我们还要**指定序列的位置position，我们约定用带尖括号的**上标来指定。



"哈利波特和赫敏格兰杰 发明了一种新的法术. "

"Harry Potter and Hermione Granger invented a new spell. "

y = [1 , 1 , 0 , 1 , 1 , 0 , 0 , 0]

1 表示这是个人名, 0 表示这不是人名 .

- 形式化表示方法:
  - 使用$X^1, X^2, X^3 \dots X^T \dots X^9$来表示输入数据
  - 使用$Y^1, Y^2, Y^3 \dots Y^T \dots Y^9$来表示输出数据
  - 使用T_x来表示输入序列的长度，$T_x = 9$.
  - 使用T_y来表示输出序列的长度，$T_y = 9$.
  - 训练数据集中第i个样本的第t个输入序列使用$X^{(I)<t>}$表示
  - 训练数据集中第i个样本的第t个输出序列使用$y^{(I)<t>}$表示
  - 使用$T_x^{(i)}$来表示训练数据集中第i个样本输入序列的长度
  - 使用$T_y^{(i)}$来表示训练数据集中第i个样本输出序列的长度

我们可以用 9 个特征集合来表示这输入的9个单词，并按序列中的位置进行索引，比如：

$x^{<1>}, x^{<2>}, x^{<3>}, x^{<4>}, x^{<5>}, x^{<6>}, x^{<7>}, x^{<8>}, x^{<9>}$

输出数据也是一样， y 表示 x 位置的输出 .

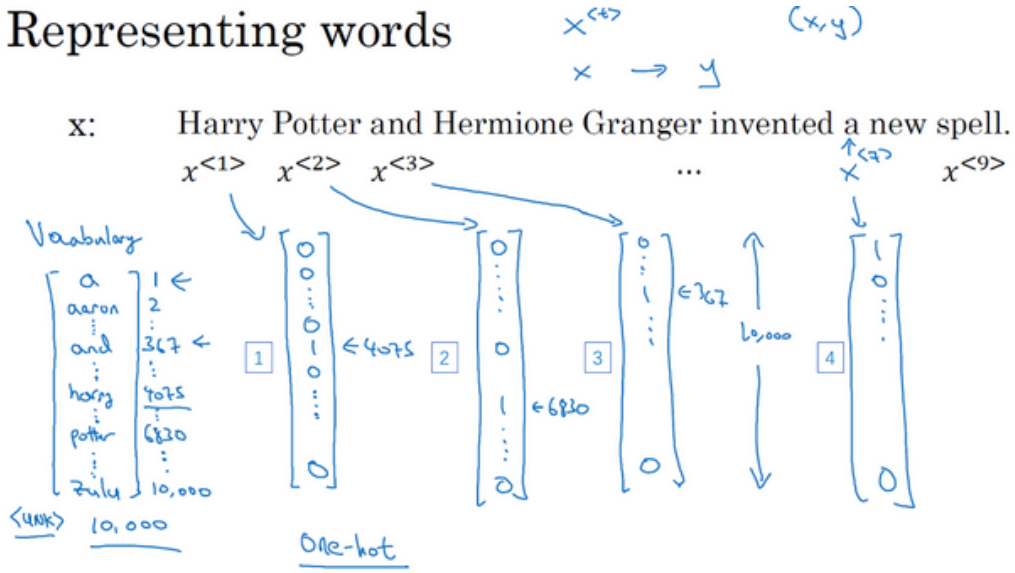
$y^{<1>}, y^{<2>}, y^{<3>}, y^{<4>}, y^{<5>}, y^{<6>}, y^{<7>}, y^{<8>}, y^{<9>}$

使用 T_x 表示输入序列的长度． T_y 表示输出序列的长度．所以， X_i 就表示第 i 条训练数据的第 t 个位置的 word．同样的， $T_x^{(i)}$ 就表示第 i 条输入训练数据的长度．

所以在这个例子中，$T_x^{(i)} = 9$，但如果另一个样本是由15个单词组成的句子，那么对于这个训练样本，$T_x^{(i)} = 15$。

既然我们这个例子是**NLP**，也就是自然语言处理，这是我们初次涉足自然语言处理，一件我们需要事先决定的事是怎样表示一个序列里单独的单词，你会怎样表示像**Harry**这样的单词，$x^{<1>}$实际应该是什么？

接下来我们讨论一下怎样表示一个句子里单个的词。想要表示一个句子里的单词，第一件事是做一张词表，有时也称为词典，意思是列一列你的表示方法中用到的单词。这个词表（下图所示）中的第一个词是**a**，也就是说词典中的第一个单词是**a**，第二个单词是**Aaron**，然后更下面一些是单词**and**，再后面你会找到**Harry**，然后找到**Potter**，这样一直到最后，词典里最后一个单词可能是**Zulu**。



举个例子，在这里$x^{<1>}$表示**Harry**这个单词，它就是一个第4075行是1，其余值都是0的向量（上图编号1所示），因为那是**Harry**在这个词典里的位置。

同样$x^{<2>}$是个第6830行是1，其余位置都是0的向量（上图编号2所示）。

and在词典里排第367，所以$x^{<3>}$就是第367行是1，其余值都是0的向量（上图编号3所示）。如果你的词典大小是10,000的话，那么这里的每个向量都是10,000维的。

因为**a**是字典第一个单词，$x^{<7>}$对应**a**，那么这个向量的第一个位置为1，其余位置都是0的向量（上图编号4所示）。

所以这种表示方法中，$x^{<t>}$指代句子里的任意词，它就是个**one-hot**向量，因为它只有一个值是1，其余值都是0，所以你会有9个**one-hot**向量来表示这个句中的9个单词，目的是用这样的表示方式表示X，用序列模型在X和目标输出Y之间学习建立一个映射。我会把它当作监督学习的问题，我确信会给定带有$(x\ ,\ y)$标签的数据。

那么还剩下最后一件事，我们将在之后的视频讨论，如果你遇到了一个不在你词表中的单词，答案就是创建一个新的标记，也就是一个叫做**Unknow Word**的伪造单词，用**<UNK>**作为标记，来表示不在词表中的单词，我们之后会讨论更多有关这个的内容。

总结一下本节课的内容，我们描述了一套符号用来表述你的训练集里的序列数据x和y，在下节课我们开始讲述循环神经网络中如何构建X到Y的映射。

03.序列模型优势

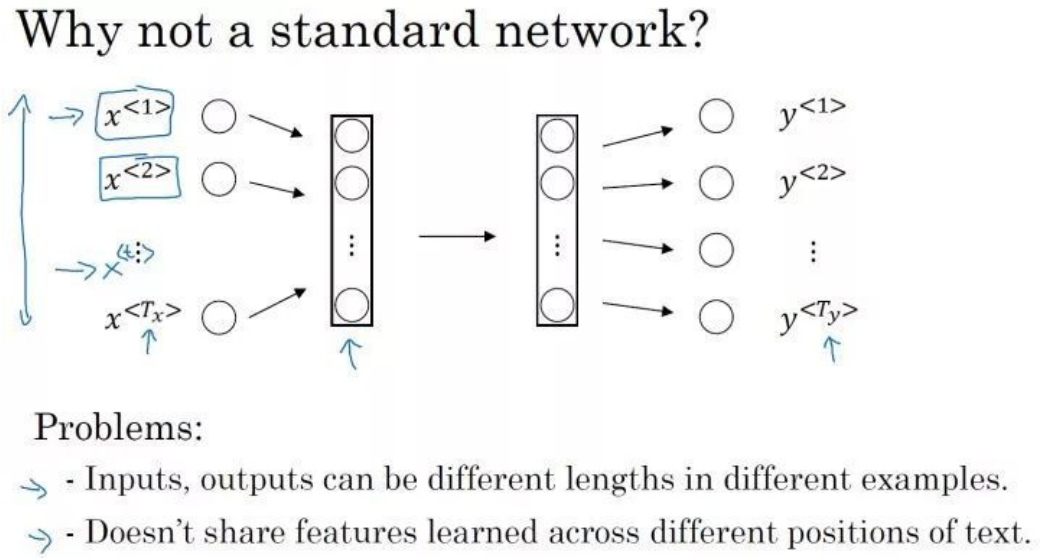
我们为什么要使用RNN这样的序列模型，而不是直接使用标准的全连接神经网络来解决输入或输出为序列数据的问题呢?

主要基于以下几点。

第一，全连接的神经网络不能够用一个模型适应输入或输出中不同的序列长度。例如，在不使用数据填充的技巧下，**无法用同一个全连接模型架构对15个单词的长度的句子和150个单词长度的句子进行情感分析。但是RNN则能够自然地适应这种序列长度的变化。**

第二，全连接神经网络不能够共享在序列不同位置学到的权重。这会导致参数过多的问题。而RNN则能够跨时间共享权重。

此外，以RNN为基础的序列模型通常还有时间平移非对称的特性，通常模型会更容易受到输入序列中较后位置的数据的影响。这一特性在时间序列预测等问题中通常是非常重要的，而全连接神经网络和卷积神经网络则不具有这样的特性。

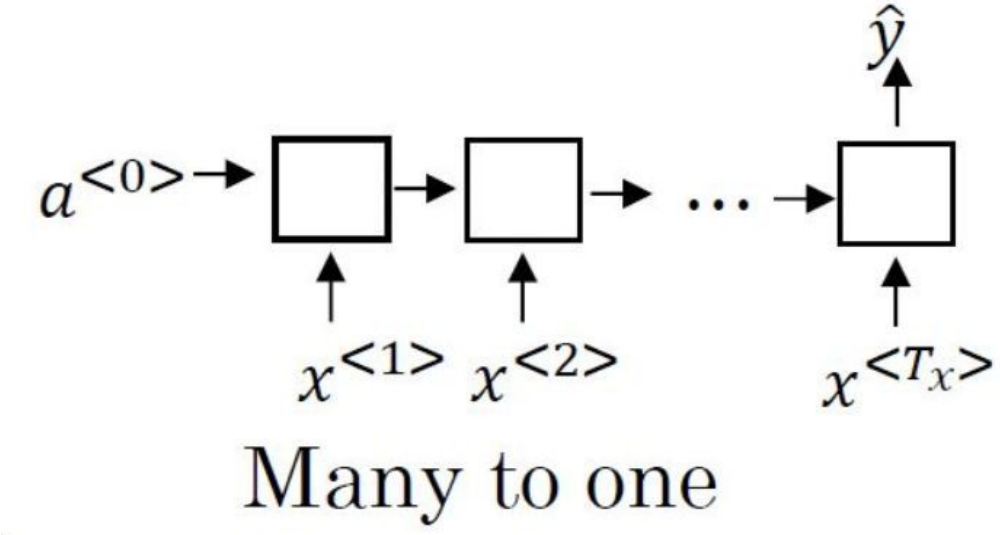


4.序列模型类型

根据输入和输出的数量及输出生成方式，我们可以把序列模型分成以下一些类型。

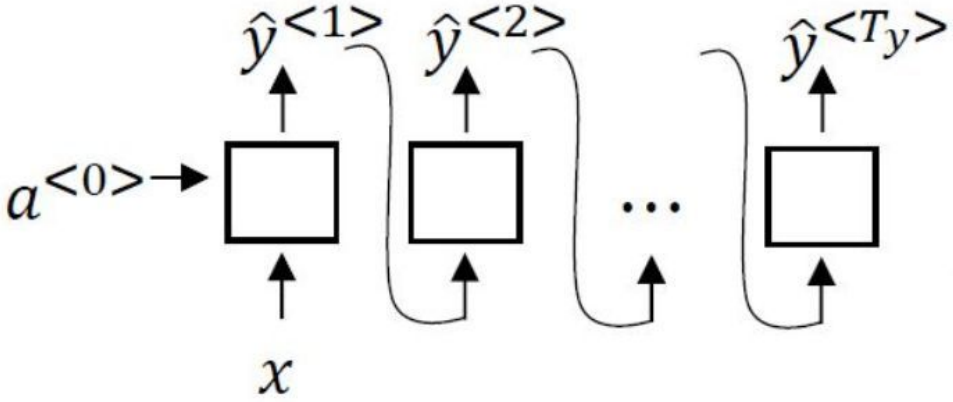
（1）Many2One

例如情感分析。我们需要对序列数据进行正负判定或者打星操作。这种情况下，**输入是一个序列，但输出是一个值。**



（2）One2Many

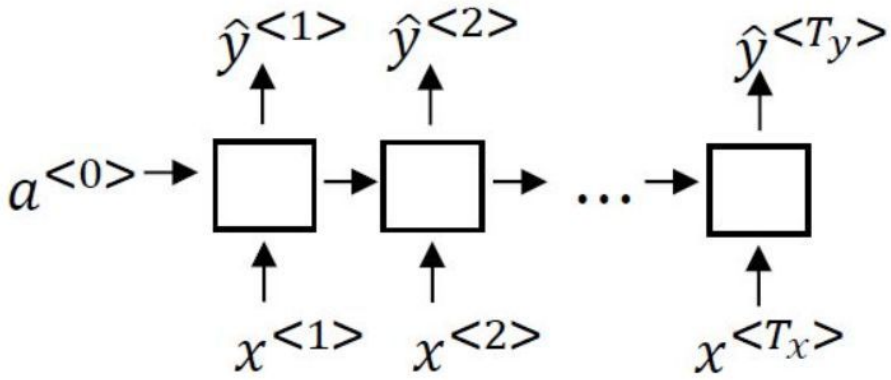
例如音乐生成。输入一个音乐的类型或者空值，直接生成一段音乐序列或者音符序列。在这种情况下，**输入是一个值，但输出是一个序列。**



One to many

（3）Many2Many

例如序列标注。我们标注一个句子中每个词是否为实体名称。这时候，输入是一个序列，输出也是一个序列，并且它们的长度是一样的。



Many to many $T_x = T_y$

（4）Seq2Seq

例如机器翻译。这也是一种多对多的结构，**但是输入和输出的长度却通常是不同的**。我们通常会使用Encoder-Decoder架构来处理这种问题。**即我们先用一个RNN网络作为编码器将输入序列压缩成一个向量，然后将压缩后的向量表示输入到一个作为解码器的RNN网络中产生输出。可以将Seq2Seq模型看成Many2One和One2Many的组合。**