智能之门

神经网络和深度学习入门

（基于Python的实现）

STEP 9 循环神经网络

第 20 章

高级循环神经网络

20.1 高级循环神经网络概述

20.2 LSTM基本原理

20.3 GRU基本原理

20.4 序列到序列

在高级循环神经网络部分，我们将介绍LSTM、GRU、序列到序列的模型的原理，为以后学习自然语言处理打下坚实的基础。

20.1 高级循环神经网络概述

循环神经网络弥补了前馈神经网络的不足，可以更好的处理时序相关的问题，扩大了神经网络解决问题的范围。

但传统的循环神经网络也有自身的缺陷，由于容易产生梯度爆炸和梯度消失的问题，导致很难处理长距离的依赖。传统神经网络模型，不论是一对多、多对一、多对多，都很难处理不确定序列输出的问题，一般需要输出序列为1，或与输入相同。在机器翻译等问题上产生了局限性。

针对上述问题，科学家们对普通循环神经网络进行改进，以便处理更复杂场景的数据的模型，提出了如LSTM, GRU, Seq2Seq等模型。

20.1 高级循环神经网络概述

- **长短时记忆网络（LSTM）**
  - 长短时记忆网络（LSTM）是最先提出的改进算法，由于门控单元的引入，从根本上解决了梯度爆炸和消失的问题，使网络可以处理长距离依赖。
- **门控循环单元网络（GRU）**
  - LSTM网络结构中有三个门控单元和两个状态，参数较多，实现复杂。为此，针对LSTM提出了许多变体，其中门控循环单元网络是最流行的一种，它将三个门减少为两个，状态也只保留一个，和普通循环神经网络保持一致。
- **序列到序列网络（Sequence-to-Sequence）**
  - LSTM与其变体很好地解决了网络中梯度爆炸和消失的问题。但LSTM有一个缺陷，无法处理输入和输出序列不等长的问题，为此提出了序列到序列（Sequence-to-Sequence, 简称Seq2Seq）模型，引入和编码解码机制（Encoder-Decoder），在机器翻译等领域取得了很大的成果，进一步提升了循环神经网络的处理范围。

20.2 LSTM基本原理

循环神经网络（RNN）的提出，使神经网络可以训练和解决带有时序信息的任务，大大拓宽了神经网络的使用范围。但是原始的RNN有明显的缺陷。不管是双向RNN，还是深度RNN，都有一个严重的缺陷：训练过程中经常会出现梯度爆炸和梯度消失的问题，以至于原始的RNN很难处理长距离的依赖。

➢ 实例角度

例如，在语言生成问题中：“佳佳今天帮助妈妈洗碗，帮助爸爸修理椅子，还帮助爷爷奶奶照顾小狗毛毛，大家都夸奖了_____。”

例句中出现了很多人，空白处要填谁呢？我们知道是“佳佳”，但传统RNN无法很好学习这么远距离的依赖关系。

20.2 LSTM基本原理

➢ 理论角度

根据循环神经网络的反向传播算法，可得到任意时刻k，误差项沿时间反向传播的公式如下：

$$\delta_k^T = \delta_t^T \prod_{i=k}^{t-1} diag[f'(z_i)]W$$

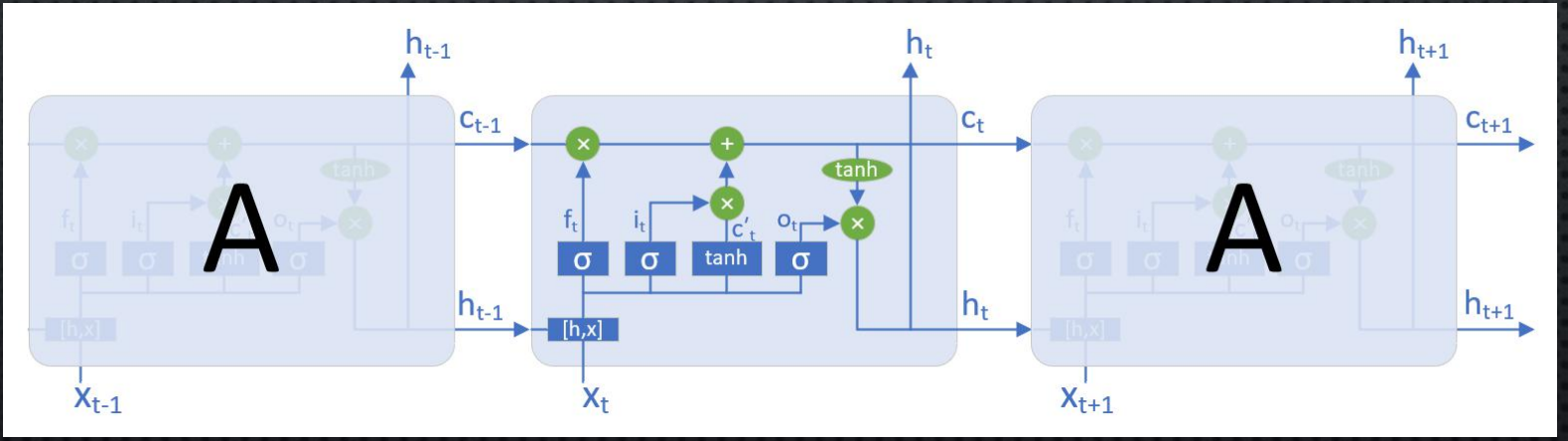
如果W的值在 $(0,1)$ 的范围内， 则随着t的增大，连乘项会越来越趋近于0， 误差无法传播，这就导致了梯度消失的问题。

如果W的值很大，使得$diag[f'(z_i)]W$的值大于1， 则随着t的增大，连乘项的值会呈指数增长，并趋向于无穷，产生梯度爆炸。

20.2 LSTM基本原理

➢ LSTM的结构

- LSTM 的设计思路比较简单，原来的RNN中隐藏层只有一个状态h，对短期输入敏感，现在再增加一个状态c，来保存长期状态。这个新增状态称为细胞状态或单元状态。
- LSTM设计了 门控结构，控制信息的保留和丢弃。LSTM有三个门，分别是：遗忘门，输入门和输出门。



20.2 LSTM基本原理

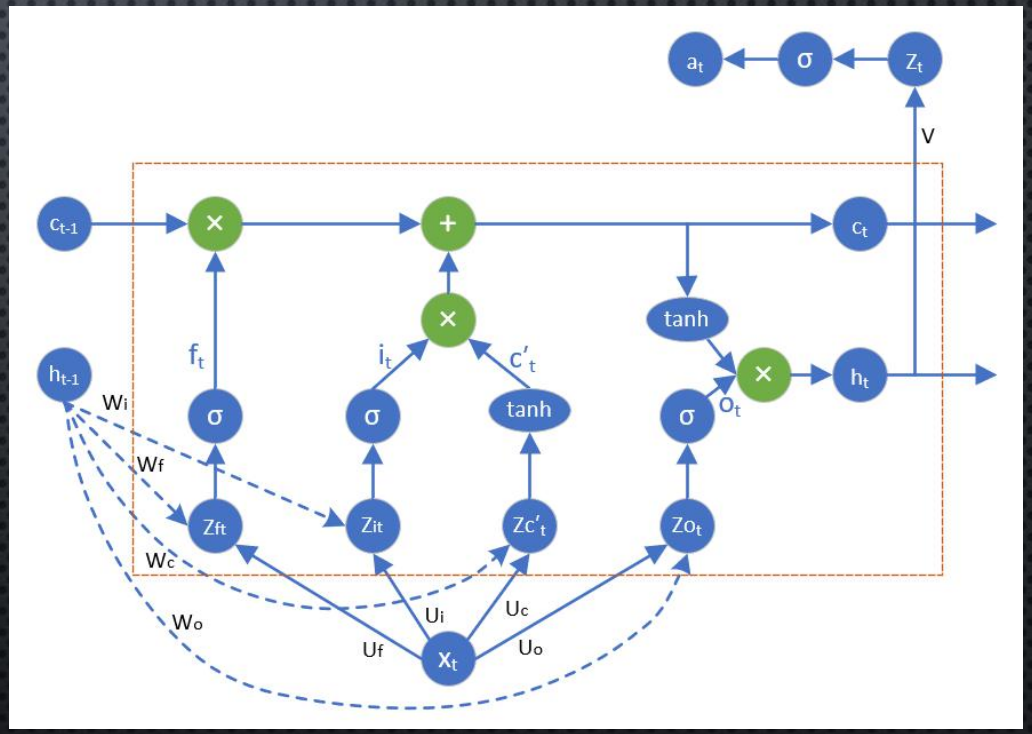
➢ LSTM的前向计算

- 遗忘门：$f_t = \sigma(h_{t-1} \cdot W_f + x_t \cdot U_f + b_f)$
- 输入门：$i_t = \sigma(h_{t-1} \cdot W_i + x_t \cdot U_i + b_i)$
- 即时细胞状态：

  $c'_t = \tanh(h_{t-1} \cdot W_c + x_t \cdot U_c + b_c)$
- 长期当前细胞状态：

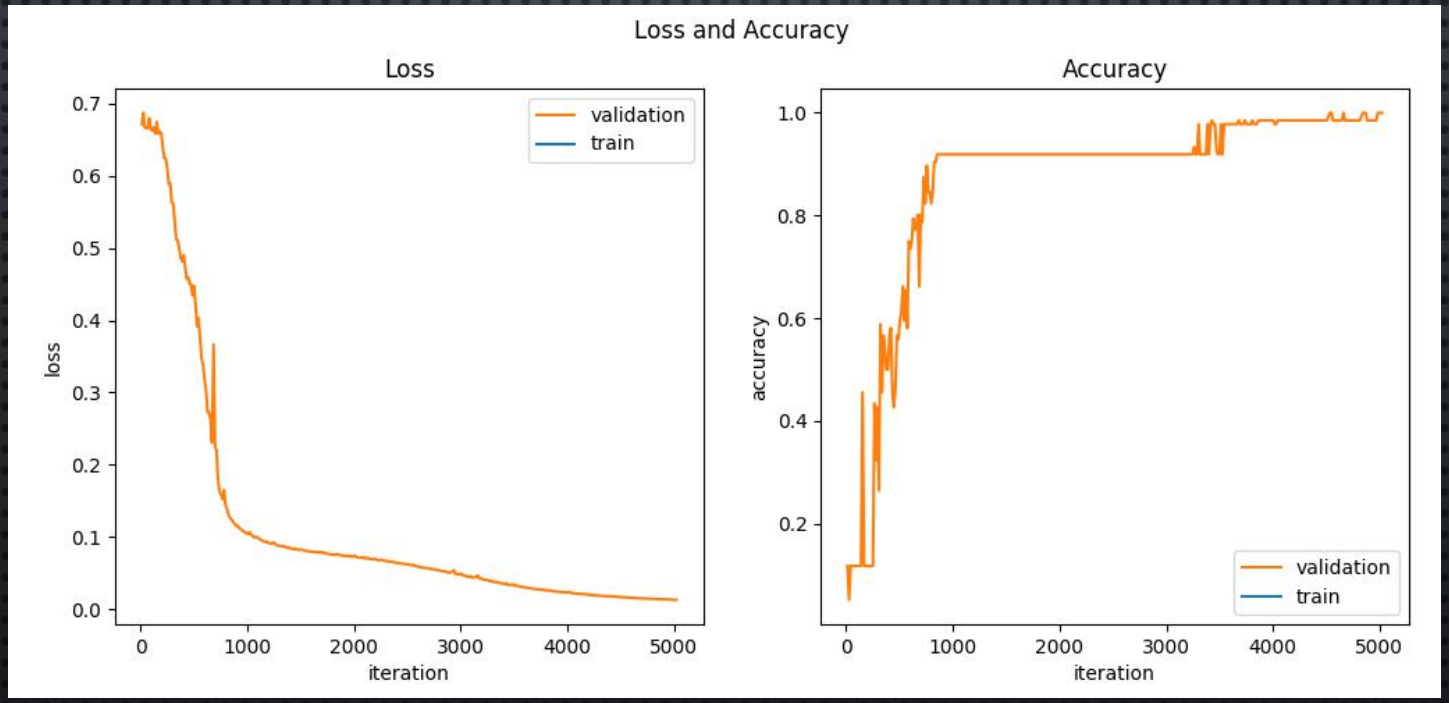
  $$c_t = f_t \circ c_{t-1} + i_t \circ c'_t$$
- 输出门：$o_t = \sigma(h_{t-1} \cdot W_o + x_t \cdot U_o + b_o)$
- 隐藏状态输出：$h_t = o_t \circ \tanh(c_t)$
- 预测输出：$z_t = h_t \cdot V + b,\ a_t = \sigma(z_t)$

➢ LSTM的反向传播



20.2 LSTM基本原理

- **LSTM的训练结果**
  - 该模型在验证集上可得100%的正确率，随机测试样例预测值与真实值完全一致。



20.3 GRU基本原理

LSTM 存在很多变体，其中门控循环单元（Gated Recurrent Unit, GRU）是最常见的一种，也是目前比较流行的一种。GRU是由 Cho 等人在2014年提出的，它对LSTM做了一些简化：

- GRU将LSTM原来的三个门简化成为两个：重置门r_t和更新门z_t。
- GRU不保留单元状态c_t，只保留隐藏状态c_t作为单元输出，和传统RNN的结构保持一致。
- 重置门直接作用于前一时刻的隐藏状态h_{t-1}。

20.3 GRU基本原理

➢ GRU的前向计算

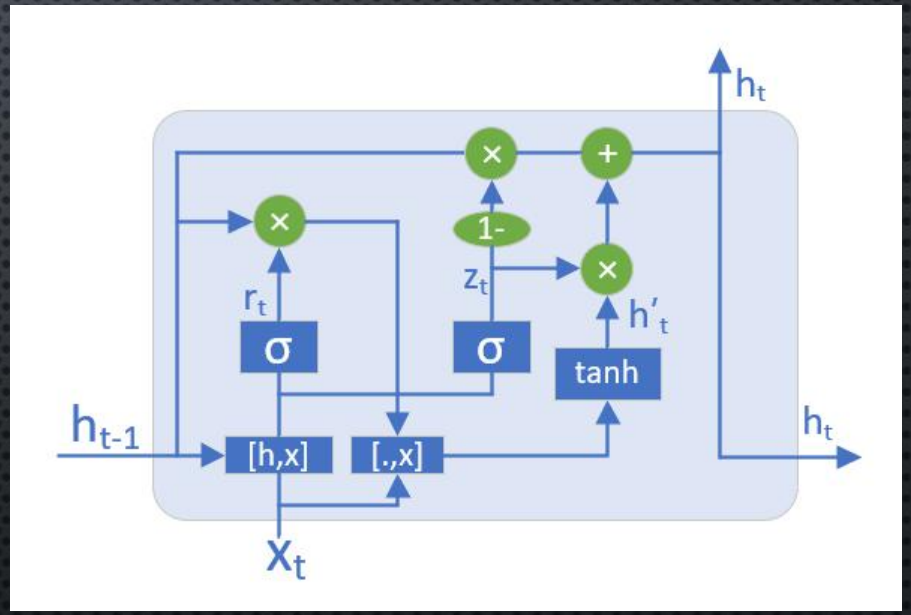
- 更新门：$z_t = \sigma(h_{t-1} \cdot W_z + x_t \cdot U_z)$
- 重置门：$r_t = \sigma(h_{t-1} \cdot W_r + x_t \cdot U_r)$
- 候选隐藏状态：

$$\tilde{h}_t = \tanh\left((r_t \circ h_{t-1}) \cdot W_h + x_t \cdot U_h\right)$$

- 隐藏状态：

$$h = (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t$$

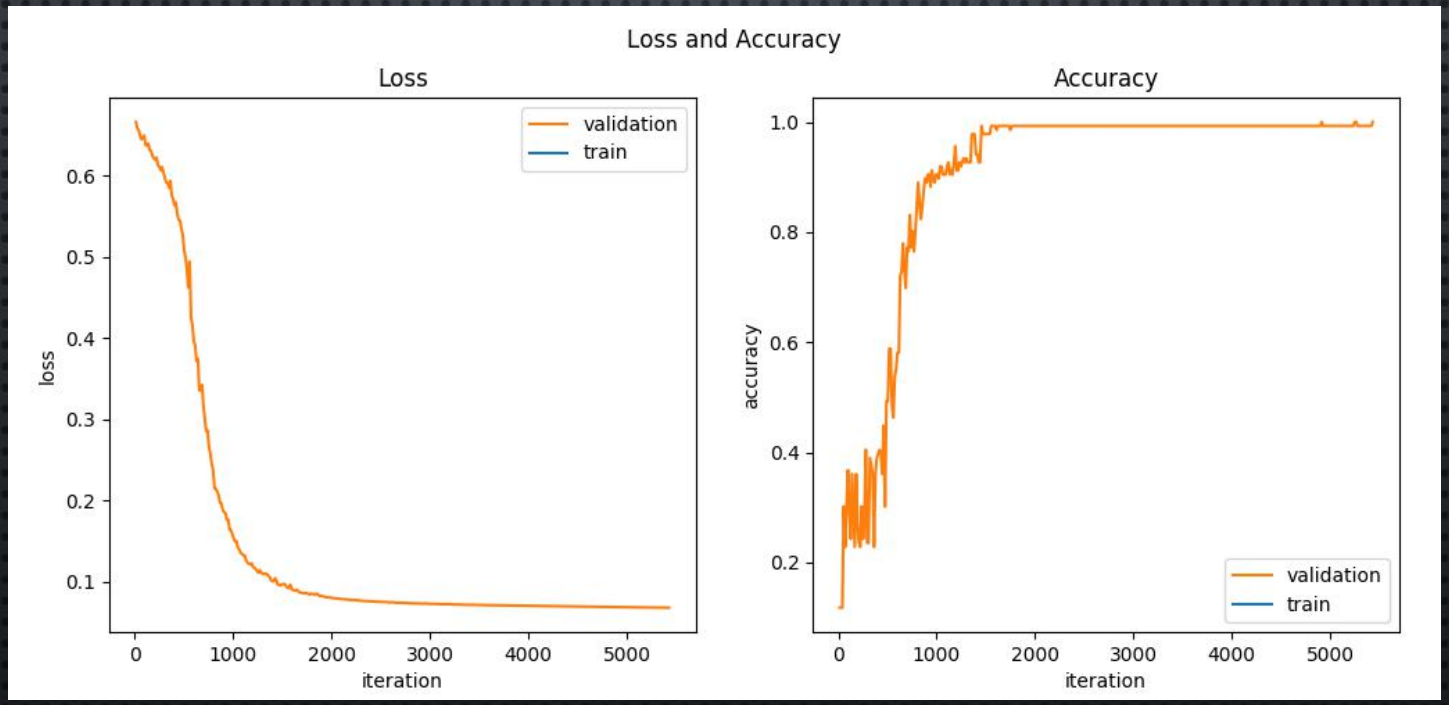
从上面的公式可以看出，GRU通过更新门和重置门控制长期状态的遗忘和保留，以及当前输入信息的选择。更新门和重置门将输入信息映射到[0,1]区间，实现门控功能。



➢ GRU的反向传播

20.3 GRU基本原理

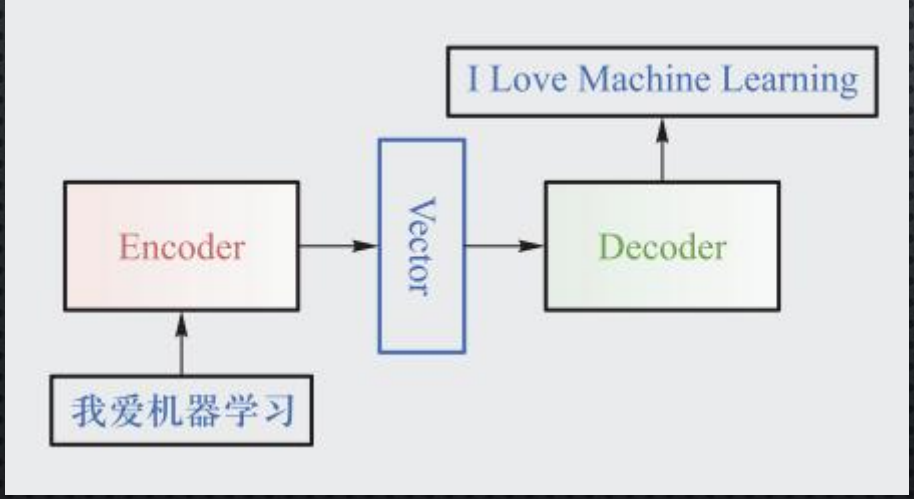
- **GRU的训练结果**
  - 该模型在验证集上可得100%的正确率，网络正确性得到验证。



20.4 序列到序列

还有一类序列预测问题，以序列作为输入，需要输出也是序列，并且输入和输出序列长度不确定，并不断变化。这类问题被成为序列到序列（Seq2Seq）预测问题。

序列到序列问题有很多应用场景，比如机器翻译、问答系统、文档摘要生成等。简单的 RNN 或 LSRM 结构无法处理这类问题，于是科学家们提出了一种新的结构——编码-解码结构。



编码器将输入序列编码成为固定长度的状态向量，通常称为语义编码向量。解码器将语义编码向量作为原始输入，解码成所需要的输出序列。

在具体实现中，编码器、解码器可以有不同选择，可自由组合。常见的选择有CNN、RNN、GRU、LSTM等。

20.4 序列到序列

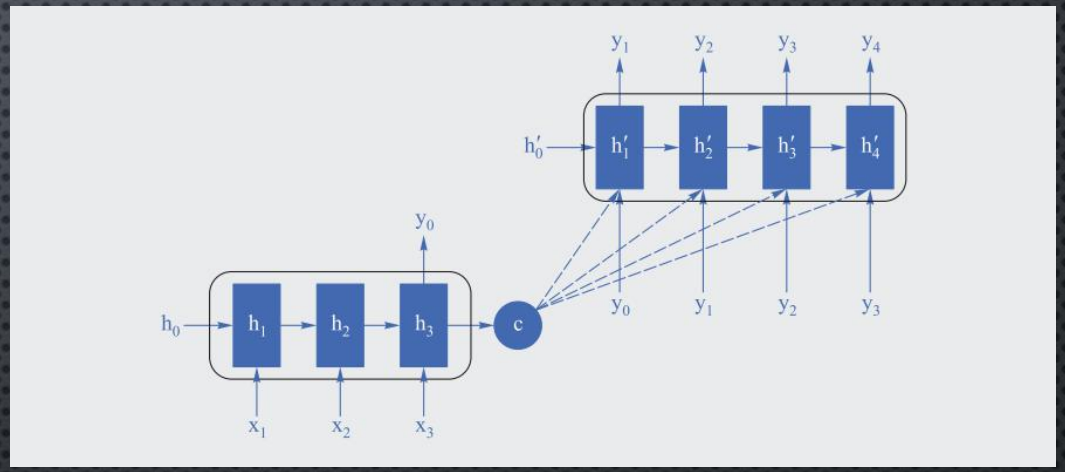
➢ 序列到序列模型

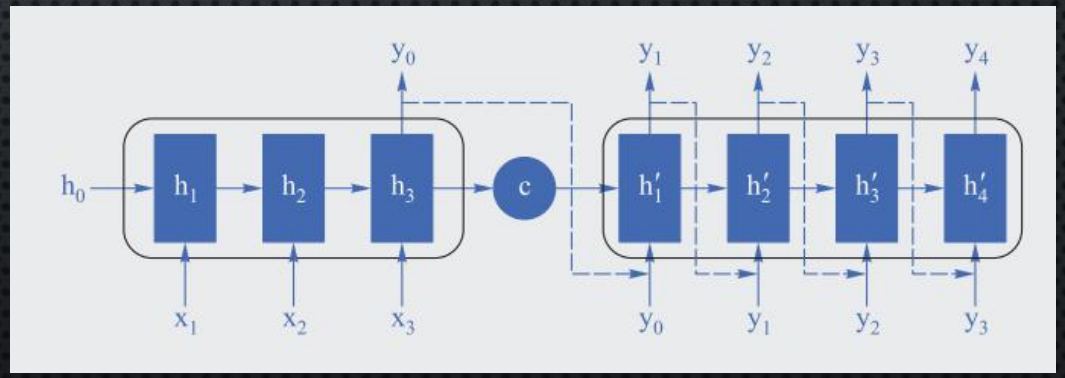
- 编码过程
  - ✓ 两种结构的编码过程完全一致。输入序列为 $x = [x_1, x_2, x_3]$。
  - ✓ RNN 网络中，每个时间节点隐藏层状态为 $h_t = f(h_{t-1}, x_t), t = 1,2,3.$
  - ✓ 编码器中输出的语义编码向量可以有三种不同选取方式，分别是：

$$c = h_3$$

$$c = g(h_3)$$

$$c = g(h_1, h_2, h_3)$$





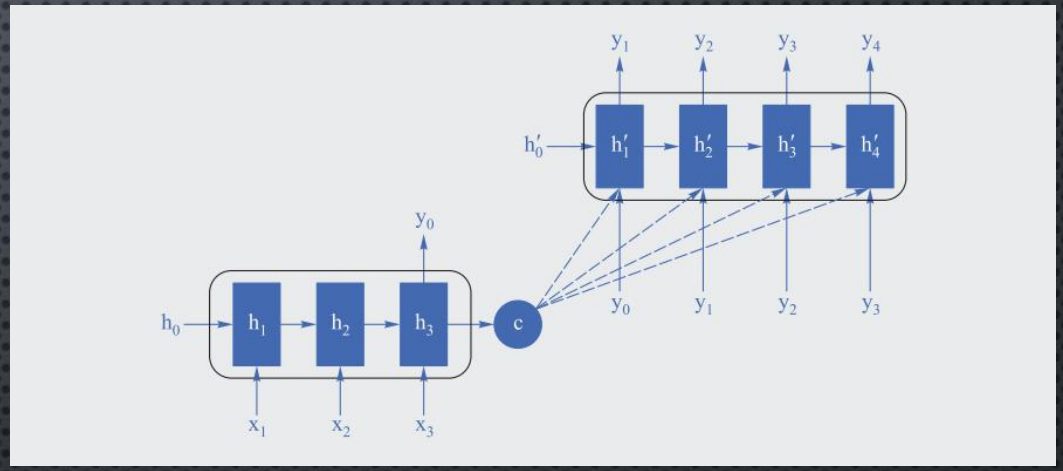
20.4 序列到序列

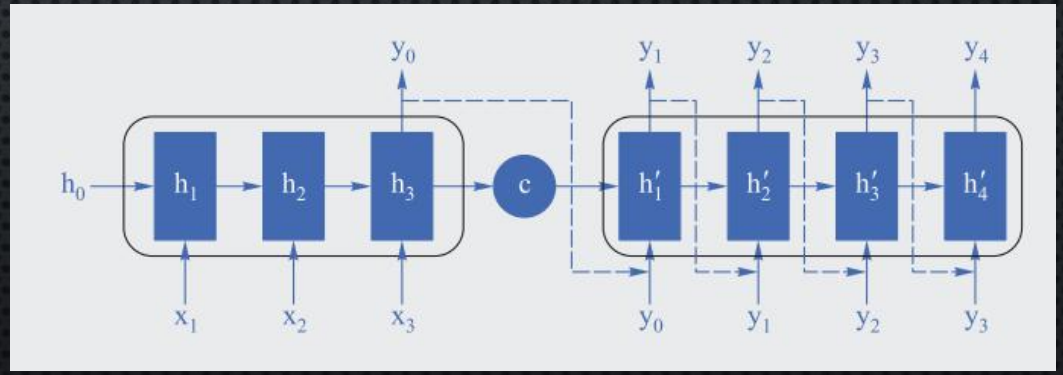
- 解码过程
  - ✓ 两种结构解码过程的不同点在于语义编码向量是否应用于每一时刻输入。
  - ✓ 第一种结构，每一时刻的输出y_t由前一时刻的输出y_{t-1}，前一时刻的隐藏层状态h'_{t-1}和c共同决定，即：

$$y_t = f(y_{t-1}, h'_{t-1}, c)$$

  - ✓ 第二种结构，c只作为初始状态传入解码器，并不参与每一时刻输入，即：

$$\begin{cases} y_1 = f(y_0, h'_0, c) \\ y_t = f(y_{t-1}, h'_{t-1}) \end{cases}$$





THE END

谢谢！