

# Recurrent Neural Network

## 循环神经网络

### 语言模型 (Language Model)

自然语言文本  $\rightarrow$  离散时间序列

- 长度为  $T$  的文本  $w_1, w_2, \dots, w_T$   $w_t \rightarrow$  时间  $t$  的输出或标签

- 语言模型计算序列概率  $P(w_1, w_2, \dots, w_T)$

$\rightarrow$  可用于提升语音识别和机器翻译的性能

$\rightarrow$  根据概率输出文本排列序列

### 语言模型的计算

假设  $w_1, w_2, \dots, w_T$  依次生成,  $P(w_1, w_2, \dots, w_T) = \prod_{t=1}^T P(w_t | w_1, \dots, w_{t-1})$

语言模型参数 计算词的概率, 以及一个词在给定前几个词情况下的条件概率

$$P(w_1, w_2, w_3, w_4) = P(w_1) P(w_2 | w_1) P(w_3 | w_1, w_2) P(w_4 | w_1, w_2, w_3)$$

$$P(w_1) = \frac{\text{词出现的次数}}{\text{总词数}}$$

$$P(w_2 | w_1) = \frac{P(w_1, w_2)}{P(w_1)} \rightarrow \text{两词相邻的概率}$$

$$P(w_3 | w_1, w_2) = \frac{P(w_1, w_2, w_3)}{P(w_1, w_2)}$$

### $n$ 元语法 ( $n$ -grams)

$n$  衡计算复杂度和模型准确性

通过马尔可夫假设简化语言模型计算,  $n$  元语法  $\rightarrow$   $n-1$  阶马尔可夫链概率语言模型

一个词的出现只与前面  $n$  个词相关  $\rightarrow n$  阶马尔可夫链

- 将语言模型改写为  $n$  元语法 ( $n$ -grams)

$$P(w_1, w_2, \dots, w_T) = \prod_{t=1}^T P(w_t | w_1, \dots, w_{t-1}) \approx \prod_{t=1}^T P(w_t | w_{t-(n-1)}, \dots, w_{t-1})$$

$$P(w_1, w_2, w_3, w_4) = P(w_1) P(w_2) P(w_3) P(w_4), \text{ unigram}$$

$$P(w_1, w_2, w_3, w_4) = P(w_1) P(w_2 | w_1) P(w_3 | w_1, w_2) P(w_4 | w_1, w_2), \text{ bigram}$$

$$P(w_1, w_2, w_3, w_4) = P(w_1) P(w_2 | w_1) P(w_3 | w_1, w_2) P(w_4 | w_2, w_3), \text{ trigram}$$

- $n$  较小时,  $n$  元语法不准确;  $n$  较大, 需要计算并存储大量词频和多词相邻概率

### 循环神经网络 (RNN)

$n$  元语法 增大  $n$ , 复杂度指数级上升

$\rightarrow$  并非刚性记忆所有固定长度的序列, 通过隐藏状态存储之前时间步的信息,

$$\left\{ \begin{array}{l} H = \phi(XW_{xh} + bh) \end{array} \right.$$

$$\left\{ \begin{array}{l} O = HW_{oh} + bo \end{array} \right. \text{ (不含隐藏状态的神经网络)}$$

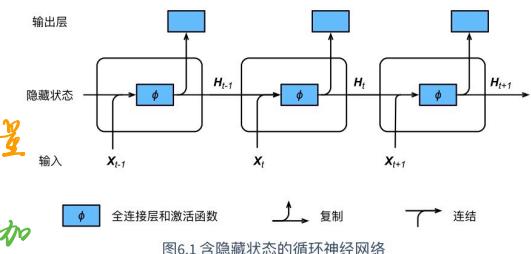
循环神经网络 上一时间的隐藏变量捕获序列截至当前时刻历史信息

$$\left\{ \begin{array}{l} H_t = \phi(X_t W_{xh} + (H_{t-1} W_{hh} + bh)) \end{array} \right.$$

隐藏变量(状态) 描述当前时间步如何上一时间的隐藏变量

$$\left\{ \begin{array}{l} O_t = H_t W_{oh} + bo \end{array} \right.$$

$\rightarrow$  不同时间步始终使用这些参数  $\rightarrow$  RNN 参数数量不随时间步增加



One-hot 向量 将词表示成向量输入到神经网络

- 词典中不同字符的数量为  $N$  (vocab\_size)

- 每个字符与索引值 (0~N-1) 一一对应 ( $i$ )

- One-hot 向量 为每个字符创建一个全  $N$  长的向量, 在位置为  $i$  处元素设为 1

### 裁剪梯度 (Clip Gradient)

应对梯度爆炸

$\rightarrow$  将所有模型参数梯度元素拼接为向量  $g$ ,

$$\text{裁剪后梯度 } \min(\lVert g \rVert_2, 1) g \rightarrow \lVert g \rVert_2 \leq 1$$

裁剪阈值

困惑度 (Perplexity) 对数熵指收敛前的值

评价语言模型的好坏

- 最佳情况下, 模型总是把标签类别的概率预测为 1, 此时困惑度为 1;
- 最坏情况下, 模型总是把标签类别的概率预测为 0, 此时困惑度为正无穷;
- 基线情况下, 模型总是预测所有类别的概率都相同, 此时困惑度为类别个数。

## 编码器-解码器 (Seq2seq) 输入并输出不定长序列

- 输入输出都是不定长序列  $\rightarrow$  encoder-decoder 模型
- 2个 RNN  $\rightarrow$  编码器 & 解码器 seq2seq 模型
- 编码器分析输入序列 生成输出序列  
每个句子后附上  $\langle eos \rangle$   $\rightarrow$  序列终止

## 解码器

最初时间步的输入  $\langle bos \rangle$   $\rightarrow$  序列开始

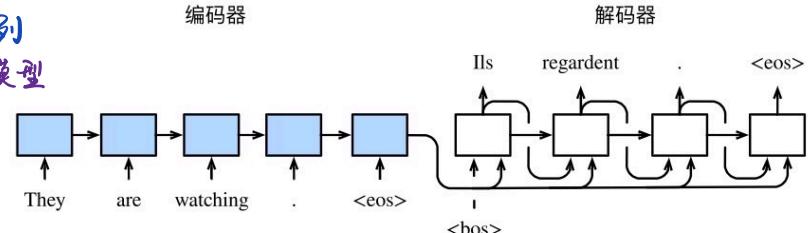


图 10.8 使用编码器-解码器将句子由英语翻译成法语。编码器和解码器分别为循环神经网络

## 编码器

- 把一个不定长的输入序列  $\rightarrow$  一个定长背景变量  $C$
- 常用 RNN 在背景变量中编码输入序列信息
- 输入  $x_1, x_2, \dots, x_T$ ,  $x_i$  为输入句子的第  $i$  个词  
在时间步  $t$ ,  $h_t = f(x_t, h_{t-1})$ , 上一时刻  
该时刻 特征向量 隐藏状态
- 编码器将各时间步的隐藏状态, 互换为背景变量  
 $C = g(h_1, \dots, h_T)$  自定义函数  
当  $g(h_1, \dots, h_T) = h_T$ , 背景变量  $C$   
是输入序列最终时间步的隐藏状态,  $h_T$

## 强制教学 (Teacher Forcing)

- 模型训练中, 所有输出序列损失的均值通常作为需要最小化的损失函数  $\rightarrow$  将上一个时间步的输出作为当前的输入
- 将标签序列 (训练集的真实输出序列) 在上一个时间步的标签作为解码器在当前时间步的输入  $\rightarrow$  强制教学

## 解码器

- 编码器  $x_1, \dots, x_T \rightarrow C \rightarrow P(y_t | y_1, \dots, y_{t-1}, C)$   
输出序列  $y_1, y_2, \dots, y_T$  在时间步  $t$ , 解码器输出  $y_t$  条件概率基于之前的输出序列  $y_1, \dots, y_{t-1}$  和  $C$
- 解码器 另一个 RNN  
在时间步  $t$ , 将上一步输出  $y_{t-1}$  与背景变量  $C$  作为输入, 与上一时刻隐藏状态  $s_{t-1} \rightarrow s_t$ . 使用函数  $g$   
 $s_t = g(y_{t-1}, C, s_{t-1})$   
 $P(y_t | y_1, \dots, y_{t-1}, C)$  自定义 Output 层和 softmax 计算

$$P(y_1, \dots, y_{T'} | x_1, \dots, x_T) = \prod_{t'=1}^{T'} P(y_{t'} | y_1, \dots, y_{t'-1}, x_1, \dots, x_T)$$

输出序列基于输入序列的  
条件概率

$$= \prod_{t'=1}^{T'} P(y_{t'} | y_1, \dots, y_{t'-1}, C),$$

$$-\log P(y_1, \dots, y_{T'} | x_1, \dots, x_T) = -\sum_{t'=1}^{T'} \log P(y_{t'} | y_1, \dots, y_{t'-1}, C),$$

该输出序列的损失

## 注意力机制 (Attention)

- 解码器在各个时间步依赖相同的背景变量  $C$  获取输入序列信息
- 当编码器为 RNN, 背景变量来自最终时间步的隐藏状态
- 注意力机制  $\rightarrow$  对编码器所有时间步的隐藏做加权平均得到背景变量

$\rightarrow$  解码器每一时间步调整这些权重 (注意力权重)

$\rightarrow$  在不同时间步分别关注输入序列中的不同部分并编码进相应时间步的背景变量  
解码器在每一时间步使用可变的背景变量  $C_t'$

计算背景变量  $C_t'$

编码器在各个时间步的隐藏状态  
解码器在时间步  $t$  的隐藏状态

$\rightarrow$  计算 Softmax 运算的输入

Softmax 输出概率分布  $\alpha_{t+1}$

$$\alpha_{t+1} = \frac{e^{e_{t+1}}}{\sum_{k=1}^T e^{e_{t+1}}}, t = 1, \dots, T$$

$$e_{t+1} = a(s_{t+1}, h_t)$$

编码器在  $t$  时  
解码器  $t+1$  状态  
隐藏状态

对各个时间步的隐藏状态做加权平均得背景变量  $C_t'$

$$C_t' = \sum_{t=1}^T \alpha_{t+1} h_t$$

## $a$ 的选择

- 若两个输入向量长度相同  $\rightarrow a(s, h) = s^T h$   $v, w_s, w_h$  都是可学习参数
- 将输入连接后通过含单隐藏层的 MLP  $a(s, h) = v^T \tanh(w_s s + w_h h)$

## 矢量化计算 (更为高效)

(一一对)

## 注意力机制

Input (查询项)  $\rightarrow$  键项 (值项)  $\rightarrow$  需要加权平均

权重来自于查询项以及与该值项对应的键项

解码器隐藏状态, 编码器隐藏状态

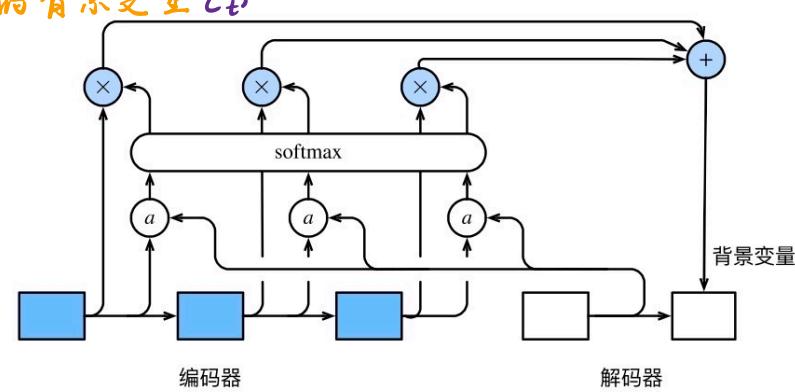


图 10.12 编码器-解码器上的注意力机制

$$s_{t-1}, h_t \in \mathbb{R}^h, t=1, \dots, T \rightarrow c_t \in \mathbb{R}^h \quad a(s, h) = s^T h$$

将查询项矩阵  $Q \in \mathbb{R}^{1 \times h}$  设为  $s_{t-1}^T$

令键项矩阵  $K \in \mathbb{R}^{Txh}$  和值项矩阵  $V \in \mathbb{R}^{Txh}$  相同, 第七行为  $h_t^T$

$$\text{softmax}(QK^T)V \rightarrow c_t^T$$

当  $Q$  行数为  $n$  将得到  $n$  行的输出矩阵, 与查询矩阵在相同行上一一对应

### 更新隐藏状态

基于 GRU1 门控循环单元  $s' = z_t \odot s_{t-1} + (1 - z_t) \odot \tilde{s}_t$

$W, b$  为门控循环单元的权重参数和偏差参数

### 变换器 (Transformer)

- Attention 能够为表征中较有价值的部分分配较多的计算资源

$\rightarrow$  Transformer

- 抛弃了 CNN 和 RNN 架构  $\rightarrow$  计算效率更高

$\rightarrow$  BERT 预训练模型

$$r_t = \sigma(W_{yr}y_{t-1} + W_{sr}s_{t-1} + W_{cr}c_t + b_r),$$

$$z_t = \sigma(W_{yz}y_{t-1} + W_{sz}s_{t-1} + W_{cz}c_t + b_z),$$

$$\tilde{s}_t = \tanh(W_{ys}y_{t-1} + W_{ss}(s_{t-1} \odot r_t) + W_{cs}c_t + b_s),$$