NLP Project1 Report

潘宜城,1901111295　　朱任博，1901210259
aqpyc@foxmail.com　　zhurenbo@pku.edu.cn

2019 年 10 月 28 日

目录

1 问题描述 **3**

2 解决方案 **3**

3 评价方式 **3**
3.1 本课程评价指标 3
3.2 更加合适的评价方式 4

4 数据分布 **4**

5 Language Model **7**
5.1 Data Representation 7
5.2 Model 7
5.3 SplitCrossEntropy 7
5.3.1 相关符号 7
5.3.2 计算过程 8
5.4 Weight Dropout 9
5.5 Embedding Dropout 10
5.6 Locked Dropout 10
5.7 Experiments 10
5.8 分析 10

6 Attention Decoder Model **11**

6.1 Corpus Data Preprocess 11

6.2 Model 12

6.3 实验 13

1 问题描述

给定故事标题，生成包含5个句子的短故事。例子如下：

Title	Five sentence Story
The Test	Jennifer has a big exam tomorrow. She got so stressed, she pulled an all-nighter. She went into class the next day, weary as can be. Her teacher stated that the test is postponed for next week. Jennifer felt bittersweet about it.
The Hurricane	Morgan and her family lived in Florida. They heard a hurricane was coming. They decided to evacuate to a relative's house. They arrived and learned from the news that it was a terrible storm. They felt lucky they had evacuated when they did.
Spaghetti Sauce	Tina made spaghetti for her boyfriend. It took a lot of work, but she was very proud. Her boyfriend ate the whole plate and said it was good. Tina tried it herself, and realized it was disgusting. She was touched that he pretended it was good to spare her feelings.

数据集包括90000个训练集和8161个测试集。

2 解决方案

从标题到故事线再到故事，前一步采用已经获得的故事线作为数据。

3 评价方式

3.1 本课程评价指标

BLEU分数，全称为Bilingual Evaluation Understudy（双语评估替换），是一个比较候选文本翻译与其他一个或多个参考翻译的评价分数。完美匹配的得分为1.0，而完全不匹配则得分为0.0。

这种评分标准是为了评估自动机器翻译系统的预测结果而开发的。虽然其在这个任务中并不是非常适用，但还是具备了以下几个优点:

- 计算速度快，计算成本低
- 容易理解
- 与具体语言无关
- 与人为评估高度相关

3.2 更加合适的评价方式

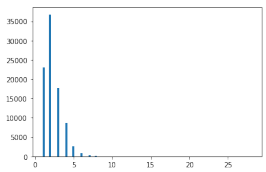
- 保真度，指故事内容与所给标题应保持一致性，即不能离题
- 逻辑性，指故事内容的叙述内容需要符合逻辑

但是，以上两种评价方式难以编码实现，其需要人为判断，故无法被采用。

4 数据分布

我们分别绘制了训练集title、测试集title和训练集sentence的长度分布。可以看出，训练集和测试集的title长度分布相似，最多的标题长度为2。训练集中，生成的故事sentence的长度可认为是一个均值为10的正态分布。

(a) 训练集title的长度分布



(b) 测试集title的长度分布

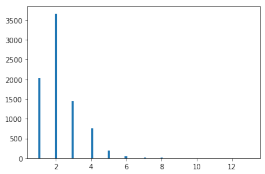
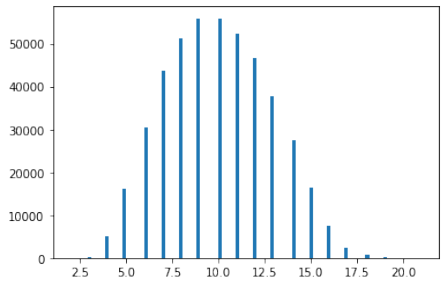


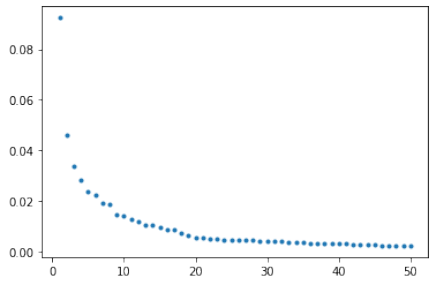
图 2: 训练集sentence的长度分布



此外，我们统计了训练集和测试集中出现的单词（包括标点）的个数及频率。

- 在训练集中，共出现了36762个单词
- 在测试集的title中，共出现了4804个单词，其中188个未出现在训练集中
- 在测试集story的grand truth中，共出现了13189个单词，其中997个未出现在训练集中

图 3: 训练集单词前50出现的频率



5 Language Model

采用语言模型的思路解决该问题。此方法先从数据中学习语言模型，也就是根据输入的词序列输出词序列的方式，然后将测试数据结合故事线输入得到测试输出。

5.1 Data Representation

本节介绍数据在模型内的表达方式，也即是模型训练的输入格式。

ROC story是由一对对的标题故事组成的数据，这里不区分它们的差别，都将其当作一段长文本的一部分，组成一个表示整个训练数据的词序列。取一组训练数据时，假设从单词w_i开始，长度为s，那么训练数据的输入为$w_i, ..., w_{i+s-1}$，输出为$w_{i+1}, ..., w_{i+s}$，即下一个词。

5.2 Model

使用的模型是一个多层的循环神经网络，输入的词序列首先通过Embedding编码，然后输入到循环神经元中产生序列输出，以真实的输出作为标签训练。网络生成测试样本的时候将前一个输出的词作为输入再次输入到模型中，循环产生整个序列，遇到终止标记字符结束。

5.3 SplitCrossEntropy

考虑到不同的词的频率不同，而在使用CrossEntropy作为损失函数的时候所有的词都将具有相同的重要性，这里根据词的频率将其划分到不同的区间，然后计算划分到较低频率区间的词的概率的时候加上对其概率的频率修正，即坟墓概率。概率修正的参数作为损失函数的可学习参数，与模型一起学习。

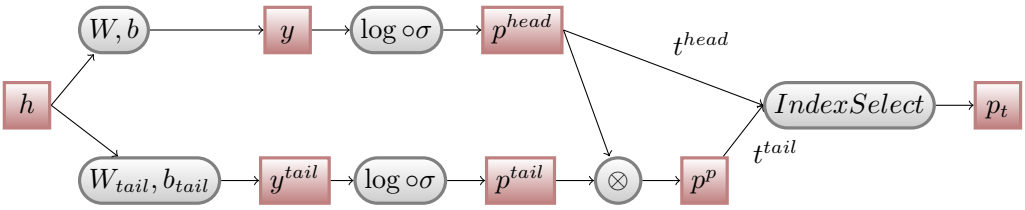
该Loss对不同区间的单词赋予不同的权重，然后再进行联合概率的计算。简单地看，对一个输出词向量，损失的计算如图所示。

5.3.1 相关符号

n_h隐藏层大小，也是循环神经网络输出的维度。

n_v词库的大小，即词的总数量，也是下面的划分的右边界，也是输出层输出的概率分布维度。

图 4: 分组交叉熵



$S = \{s_0, s_1, ..., s_{m-1}, s_m\}$用于将词分为不同分组的分割点，以词的字典索引为准。划分区间个数为m。为了包含整个词库，其中s_0为0,s_m为词种类的总数。

$W_{tail} \in R^{(m-1)\times n_h}, b_{tail} \in R^{m-1}$是损失函数的参数，用来计算词分类为坟墓词的概率。

5.3.2 计算过程

这里根据隐藏层输出H,目标词序列的索引t，输出层权重$W \in R^{n_v \times n_h}, b \in R^{n_v}$进行计算。

首先将H变形为$R^{n \times n_h}$，然后根据目标词的索引t_i和分组参数$S = \{s_0, s_1, ..., s_{m-1}, s_m\}$进行分组。结果表示为$\tilde{H} = \{\tilde{H}_1, ..., \tilde{H}_m\}, \tilde{t} = \{\tilde{t}^1, ..., \tilde{t}^m\}$。例如，$\tilde{t}^i$包含了所有目标词索引中索引在范围$[s_{i-1}, s_i)$内的词，$\tilde{H}$也是同样分组的。每个分组内词的数量记为$n_i$。

然后计算头部分组$[s_0, s_1)$的头部词概率以及坟墓的概率。这个过程使用的权重是如下组合得到的:

$$W_{head} = \begin{pmatrix} W[s_0, :] \\ W[s_0 + 1, :] \\ \vdots \\ W[s_1 - 1, :] \\ \hline W_{tail} \end{pmatrix}, b_{head} = \left(b_{s_0} b_{s_0+1} \cdots b_{s_1-1} \middle| b_{tail} \right)$$

同理使用$\tilde{H}$在行上连接，得到

$$\tilde{H}' = \begin{pmatrix} \tilde{H}_1 \\ \tilde{H}_2 \\ \vdots \\ \tilde{H}_m \end{pmatrix}$$

然后根据上述参数得到结果

$$Y^{head} = \tilde{H}' \times W_{head}^T + b_{head}$$

再对Y使用对数Softmax得到对应的概率。

$$P^{head} = \log\left(Softmax(Y^{head})\right)$$

对于属于头部词，他们的交叉熵直接为对应的项的和，如下:

$$E_{head} = -\sum_{i \in [0,\, n_1)} P^{head}_{i,\tilde{t}_i^1}$$

而对于其他分组的词，概率需要包括坟墓的概率。假设我们考虑分组$p(p = [2, 3, ...m])$，它所包含的词索引的范围是$[s_{p-1}, s_p)$，包含的词在坟墓概率P^{head}中的索引在$[i_p, i_p + n_p)$。

$$\begin{aligned} P^{p\prime} &= \log\left(Prob(Tomb_p) \times Prob(Words|Tomb_p)\right) \\ &= \left(\log\left(Prob(Tomb_p)\right) + \log\left(Prob(Words|Tomb_p)\right)\right) \\ &= P^{head}[i_p : i_p + n_p, -(m + 1 - p)] + P^p \end{aligned}$$

接着使用输出层权重中和该分组对应的权重$W[s_{p-1} : s_p, :], b[s_{p-1} : s_p]$计算词在坟墓内的分布概率$P^p$。假设$Y^p$表示线性组合的输出值。因而就有

$$\begin{aligned} Y^p &= \tilde{H}_p \times W[s_{p-1} : s_p, :]^T + b[s_{p-1} : s_p] \\ P^p &= \log\Big(Softmax(Y^p)\Big) \end{aligned}$$

所以分组p的交叉熵为

$$E^p = -\sum_{i \in [0, n_p)} P^{p\prime}_{i,\tilde{t}_i^p}$$

综上，总交叉熵H等于$\frac{1}{n} \times \left(E^{head} + \sum_p E^p\right)$.

5.4 Weight Dropout

Weight Dropout是用在循环神经网络内部的正则方式，通过随机去除循环神经元中隐藏层到隐藏层的连接，防止循环神经网络的过拟合。

5.5 Embedding Dropout

Embedding Dropout是嵌入层采用的正则项。通过随机去除输入序列中某些词的嵌入输出，即消除这个词，进行正则。

5.6 Locked Dropout

Locked Dropout是对序列的向量采用相同的Dropout的正则方式。考虑到序列中每个向量之后都会经过相同的运算，采取保持一致的Dropout有利于模型利用相同的信息。

5.7 Experiments

本节介绍采用的实验的参数和结果。

Model	Em size	Hidden size	Test ppl	Bleu 4
model tied with wdrop [1]	800	800	22.85	0.4074
model tied without wdrop	800	800	27.22	0.3908
model tied with wdrop	800	800	35.28	0.3764
model tied data fix	400	400	27.43	0.3479

[1] 使用提供的Violet代码(pytorch 0.4.0)，其他均为自己实现的代码

5.8 分析

通过以上实验结果可知，提供的代码效果最好，自己的代码是在其基础上修改而来，效果有些下降。同时也可以看到经过频率修正的数据训练的效果也不太好，这里由于没有充分的时间使用和原模型一致的参数，难以确认它有没有效果。

在测试集上频率修正后的ppl基本和最好的结果一致，说明它可以在更小的模型上达到同级别的效果，算是对分组交叉熵的效果的验证。

6 Attention Decoder Model

参考NLP FROM SCRATCH: TRANSLATION WITH A SEQUENCE TO SEQUENCE NETWORK AND ATTENTION实现Decoder中的注意力机制。这里的数据模型是序列到序列，通过限制序列的长度为定长，则可以使用注意力层对输入序列的输出进行加权，然后再依此生成目标的序列。

6.1 Corpus Data Preprocess

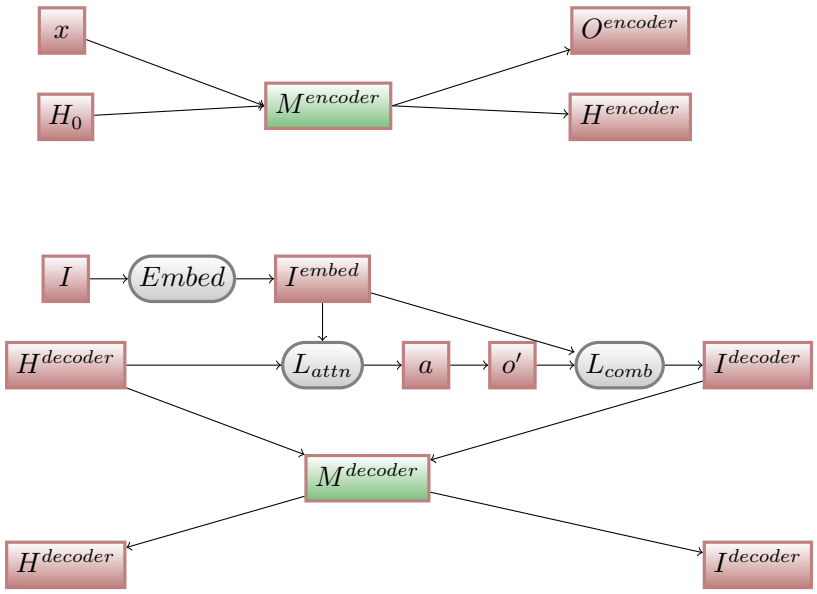
本节介绍对ROC Stories数据预处理成适合模型训练的过程。

由于注意力机制是针对序列的词施加注意力权重，这里的训练数据应该以整个序列的方式输入，同时输出也应该是同样的整个序列。考虑到ROC Stories的数据特征，这里将生成好的故事线作为输入，故事作为输出。为了使用批量训练加速训练过程，需要对不同长度的输入数据进行长度对齐操作，输出也是。若用S_{max}表示最长序列长度，N表示批量的个数，x_i^b, y_i^b表示$Batch\ b$中输入和输出序列的第i个词，那么对齐处理后的数据会具有如下矩阵所示的形状。

$$X = \begin{bmatrix} x_0^0 & x_0^1 & \cdots & x_0^{N-1} \\ x_1^0 & x_1^1 & \cdots & x_1^{N-1} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \end{bmatrix} Y = \begin{bmatrix} y_0^0 & y_0^1 & \cdots & y_0^{N-1} \\ y_1^0 & y_1^1 & \cdots & y_1^{N-1} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

6.2 Model

图 5: 注意力机制模型



由于数据已经对齐，介绍模型时采用一组数据以方便符号的使用。记$x_1, ..., x_s$和$y_1, ..., y_s$分别为输入输出的序列。输入编码器为一个循环神经网络$M^{encoder}$，经过对输入编码得到$O^{encoder}, H^{encoder} = M^{encoder}(x_1, ..., x_s, H_0)$。

解码器阶段通过循环输入实现，输入表示为$I, H^{decoder}$，这里的初始输入为起始字符$< SOS >$的词索引，其中$H^{decoder}$的初始值为上面编码器的输出隐藏状态$H^{encoder}$。

输入经过Embedding得到词向量$I^{embed} = Embed(I)$，然后生成注意力权重

$$a = \sigma(W_{attn} \begin{bmatrix} I^{embed} \\ H^{decoder} \end{bmatrix} + b_{attn})$$

应用注意力权重到编码器的输出上，得到加权后的输出，$o' = a \odot O^{encoder}$。然后就可以计算解码器循环神经元的输入了，

$$I^{decoder} = \sigma(W_{combine} \begin{bmatrix} I^{embed} \\ o' \end{bmatrix} + b_{combine})$$

最后通过循环神经元得到输出的词向量

$$O^{decoder}, H^{decoder} = M^{decoder}(I^{decoder}, H^{decoder})$$

每次循环时将输出$O^{decoder}$记录下来，最后在通过输出层就可以得到具体的词概率分布了。循环到下一步时根据此概率分布采样下一个输入的词作为新的输入。

6.3 实验

这个模型由于设计比较晚，而且可能写的不够优化，没有训练完。从初期的几个周期的损失上看能够逐渐的下降，只是要完成充分训练还需要大量时间。