计算语言学作业6

陈禹东 计研三二 2018210965

一、实验内容

1. 实验内容：

基于RNN语言模型，计算句子的概率。

实验测试不同LSTM隐状态大小（hidden\_size）超参数下的结果。

实验数据（北大《人民日报》标注语料库）分为训练集、验证集和测试集，验证集、测试集用于计算所有句子的perplexity的平均值（验证集用于确定最佳超参数）。

其中为一个长度为N的句子。

2. 实验设定：

(1) 分句方式：按句末标点(。！？)分句，并将句末标点用<eos>代替（句中的标点符号也算一个词）。

(2) 单词处理：删去了与某个词相连的标点，如“[新华社”处理为“新华社”。不保留词性信息，不同词性的同一词汇看作一个词。

(3) 基于tensorflow神经网络框架：tensorflow1.5.0+cuda9.0+cudnn10.0。显卡：GeForce GTX 660 Ti。

(4) 参考代码：<https://www.tensorflow.org/tutorials/sequences/recurrent>。

二、实验步骤与原理

1. 数据预处理

首先，依据训练集语料构建单词库。

训练集里一共有49003个不同单词，为了减少神经网络的参数，缩短训练时间，我们将频次<6的单词替换为<unk>（数量占比7.3%），剩下11250个不同单词，作为单词库（于是vocab\_size = 11250）。

然后，对于验证集和测试集，我们将频次<6或未在单词库中出现的单词都替换为<unk>。

由三个数据集分别生成三个.txt文件，文件的每一行包含一个用空格分词的句子。

2. 神经网络的输入和label

为使RNN的训练过程更加可控（Back Propagation更易于计算），本实验采用截断式BP的static RNN（unrolled RNN），也就是说，LSTM的输入数和输出数始终是固定的num\_step个。不妨设num\_step=5。

（static RNN示意图）

于是，训练的大致过程就是：每轮训练（epoch）的每一步（step）前向传播过程中，我们都需要向LSTM投喂num\_step=5个单词作为输入，得到5个预测输出（预测下一个单词在词库所有单词上的softmax概率），并对应地提供5个label与预测输出作比对，计算loss，再通过BPTT（Back Propagation Through Time）计算梯度，并更新网络参数。

考虑到minibatch训练策略（不妨设batch\_size=20），我们每个step则需要投喂batch\_size\*num\_step=20\*5个单词作为输入，并对应提供20\*5个label。

为了满足这个需要，我们需要对预处理好的.txt数据稍作变换。

首先，我们将所有的换行符用<eos>替代，然后把所有句子依次相连，得到长度为data\_len（不妨设=20000）的一长串单词串。然后，我们将它等分为batch\_size =20段，每段有batch\_len=20000/20=1000个单词。对于每一段，我们再将它等分为长度=num\_step=5的很多个小段。于是，如图所示，第一步前向传播要投喂的，就是黄色区域的20\*5个单词，斜线区域为20\*5个label。

（示意图）

第二步前向传播时，则使用第二个20\*5的单词矩阵作为输入，以此类推，直到把所有的数据集单词都过一遍，完成一轮训练。

值得注意的是，Embedding

3. Loss与perplexity

二、实验结果和分析

1. 实验结果：

|  |  |  |
| --- | --- | --- |
| Model | Valid set | Test set |
| HMM (forward) | 438.82 | 434.71 |
| HMM (backward) | 438.82 | 434.71 |
| Markov (unigram) | 10257.83 | 7508.20 |
| Markov (bigram) | 9313.86 | 9048.78 |

(3) 平滑方法：adding one smoothing。

2. 实验分析：

首先，perplexity越小，句子的概率越大（perplexity的定义已消除句子长度的差异）。而在其他条件都相同的情况下，对于同一个句子，若采用方法A和方法B估计的句子概率为PA和PB，且PA>PB，则可以认为A是一个更好的估计方法，它让我们对句子的估计更加确定。

基于这个原则，我们可以发现：

(1) HMM问题1的前向、后向算法在验证集、测试集上得到的结果一致。这是理论导致的必然结论，因为前向、后向算法只是在递推方向上有差别，其求和式子展开后完全一致。

(2) HMM模型相比n-gram（Markov）模型，能更好地估计句子的概率。我们可以推测，这是因为HMM模型相比n-gram模型额外考虑了词性信息，并统计了词性（隐层）与单词（输出层）的概率关联，更充足的信息给予了它更好的估计能力。