**RNN、Embedding作业与心得**

**一、tinymind任务地址：**

**二、Embedding作业：**

通过word2vec对宋词进行学习，学习语句中前后任意两个字之间的相似度，若认为两个文字结伴出现的频率越高，其具有更高的相似度，则认为其可以用相似的向量进行表示，如饕餮其向量应该基本相同。本次作业中，采用128维的向量对其中出现频率最高的5000个文字进行表示。结果显示语义相近的字确实具有更高的相似度。

通过向量的方式表示汉子可以有效地控制数据地维度，相较于采用5000维地onehot向量，128维的向量，明显可以使输入rnn地数据大小明显降低，提高计算速度，提高神经元地利用率。

**三、RNN神经网络作业：**

rnn是一个空间方向权值共享，同时在时间方向上具有深度的神经网络，且时间方向神经网络的层数与num\_step保持一致(即每次在时间方向滑动num\_step个层数，然后再对此num\_step层数的参数进行反向传播)。如此可以学习文字之间的前后关系。Num\_step的数量越大，其学到的语义就应该越有意义，但同时若时间上的层数过多，会引起沿时间轴上的梯度消失的问题。

**util代码：**

用于读取文本，并将其转化为128维的embedding向量表达。同时将输入数据按batch\_size=32的大小分为32个数据区，然后分别从这32个数据区中依次取出数据进行训练，以保证输入数据的连续性。

**model代码:**

构建三层、隐层数量为128的rnn网络，施加dropout，将输入层和LTSM层的权重合并成一个矩阵，一次性与state和data进行计算。然后将得到的计算结果以dim\_embedding的长度全部展开一次性与最后输出层的权重进行计算得到logits。

**Train代码：**

通过接受数据生成训练所需要的feed\_dict。其中应包含输入数据、标签、state(注意第一次传入的是0矩阵)和dropout的系数。

**四、心得**

1、在embedding中遇到了如下问题：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 错误提示 | 原代码 | 修改代码 |
| 1 | TypeError: sequence index must be integer, not 'slice' | if data\_index == len(data):  buffer[:] = data[:span]  data\_index = span | if data\_index == len(data):  data\_index = 0   buffer.extend(data[:3]) |

2、在rnn中，最终训练的结果效果不理想，在30个epoch后仍有部分文字是以UNK的形式表现了出来，估计可能放入了一个embedding训练的次数不够的文件，故而更新了embedding文件并调大了batchsize数量,结果发现效果并不明显仍然出现了不少UNK，估计是否应该进一步增大epoch的数量，并将5000个常用词进行扩充？