Lab2 Part2

19302010074 张诗涵

**一、CRF Model结构(及实验过程)**

Model结构：

Dataset用于存储所有用于训练的语句，dataset\_answer保存正确结果，以嵌套列表的形式存储

U\_templates与b\_templates分别保存了unigram与bigram的特征，generate\_key\_u与generate\_key\_b根据提供的模板、模板序号、当前string与字生成特征key，并在model中通过score\_map的字典保存weight

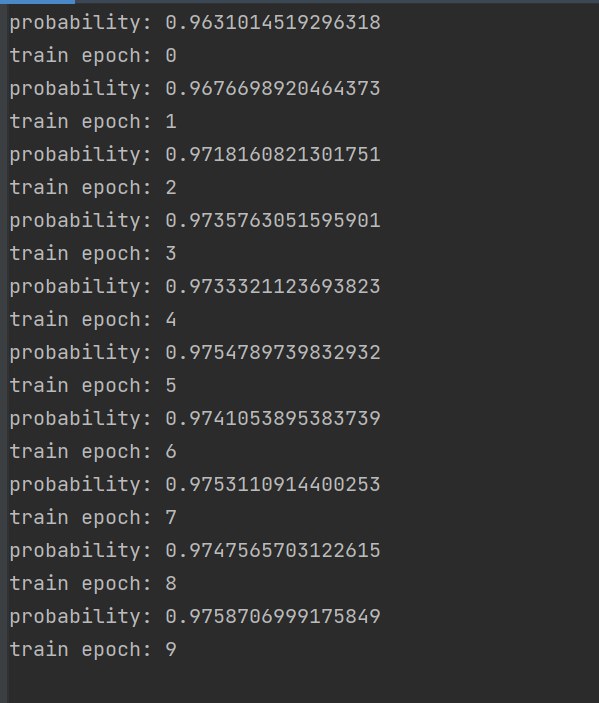
训练过程：对每个句子根据score\_map计算其最大可能路径，将最大可能路径与正确答案进行比较，在某一位上如果一致就不做修改，如果不一致则将模型打出的label根据每个template生成bad\_key, 如果score\_map中已有这个key，则将其对应的weight-1，如果没有则将其加入score\_map, weight设为-1；将正确的label根据template生成good\_key, 如果score\_map中已有则对应weight+1，如果没有则加入score\_map, 初始weight设置为1

实验中我一共踩了五个坑，其中三个源于对crf模型片面不正确的认知，最后一个尚不清楚原因：

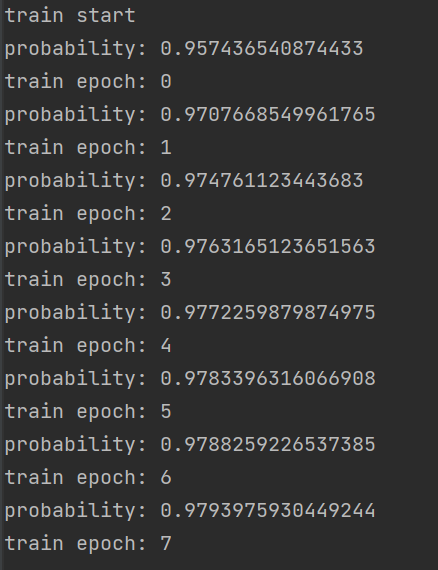
1. 误以为train.utf8中所有的句子应该拼在一起测试，但其实每句话应该分开训练，前者会导致作为开头的情况考虑只有最开始的那一种（“（完），SSS”）严重影响在测试集上的性能。
2. 对于特征函数生成的方式不了解，误以为一开始要将每个字对应的每个情况全部生成成特征函数，造成了大量无用的特征影响模型的准确率、同时也使调整更加困难
3. 最过分的错误：完全错误理解了生成path的过程，以为每次是计算出当前字每个状态的可能性然后选出最大的，再据此选择下一个即可，但事实上这每次选出的都是局部最优解、最终会错过全局最优解，应该算出全部概率后反向解码，即先解出最后一个再向前倒推。
4. 还是特征函数key的生成，一开始没有将模板序号加入key，这会导致一个问题：“我我我喜欢你”这句话，在“喜”这个字时，%x[-1,0] 和 %x[-2,0]得到的都是“我\_B”这个key，会更新在同一个key上，造成不良的影响，应该将template序号也加入考虑，果然修改后正确率大幅提高。
5. 还未发现原因的问题：原本是将几百句话全部计算出每个的结果后，再依次迭代每句话调整权重，结果就是正确率会骤降、并维持在39%的正确率训不动；改成算完一句话之后就立刻调整就解决了这个问题。

**二、在训练集上的收敛情况**

如图所示，对于dataset1，将前80%作为训练集，后20%作为测试集的收敛情况

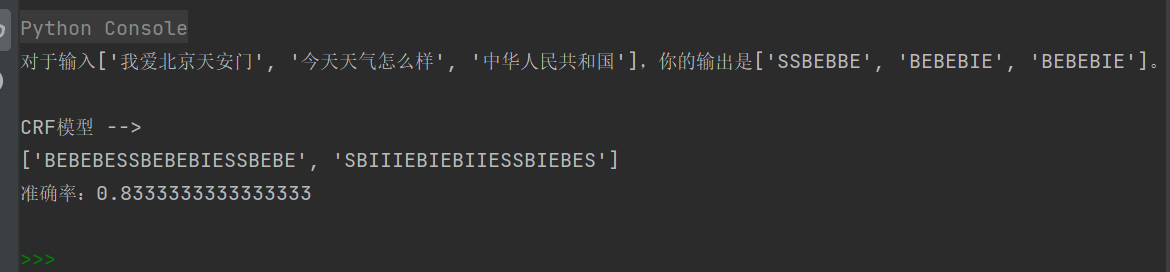


如图所示，对于dataset2，将前80%作为训练集，后20%作为测试集的收敛情况

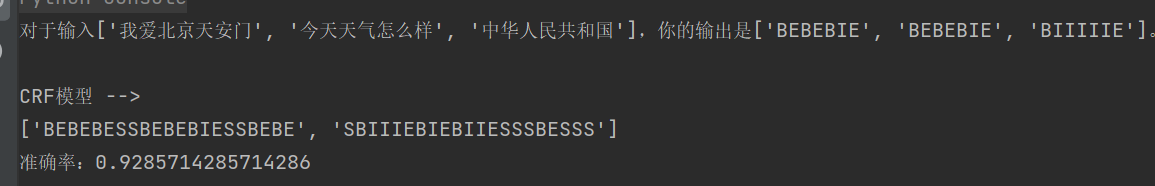


**三、在check函数中测试结果**

使用dataset1训练出的模型在check函数中跑出的结果如图：



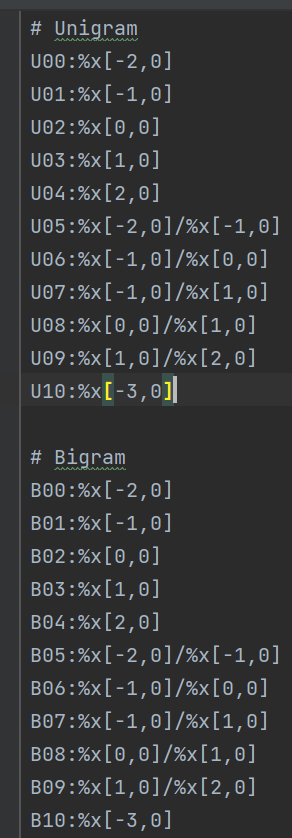
使用dataset2训练出的模型在check函数中跑出的结果如图：



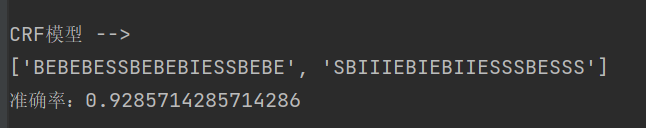
**四、自行调整模型后得到的结果**

（以上两板块均只列举了原型模板训练出的模型的收敛与测试效果，优化部分见此板块）

在模板中加入自行设计的U10与B10后，对训练效果较差的dataset1训练出的模型而言，准确率有了明显的提高，但对本身训练效果较好的dataset2训练出的模型则没有明显的帮助。







**五、BiLstm+CRF模型情况**

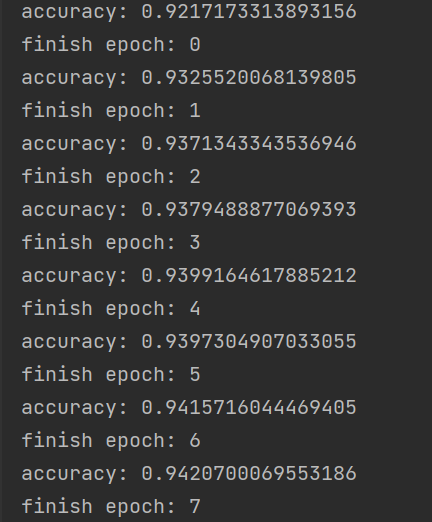
1. 模型实现的原理：

BiLstm+CRF的模型总共分为两层，即Bilstm层与CRF层，它首先将一个输入句子x经过embedding层将每个词语或字符映射为一个词向量或字符向量，然后传入BILSTM层，获得句子的前向和后向向量，接着将前向和后向向量进行拼接作为当前词汇或字符的隐藏状态向量；但如果只使用纯BiLstm其学习效果一般，这是因为它在标注时仅仅考虑了当前字符而没有考虑前后字符的状态信息，例如B标签之后是不可能跟S标签的、I标签之后也不可能跟S或B标签。因此CRF层就是起了这个约束作用，使得标注的准确率得以上升。

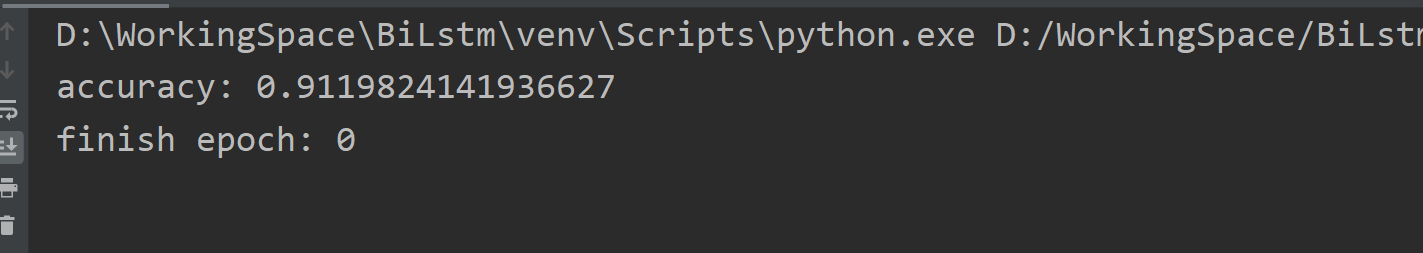
1. 训练效果：

通过更换不同的embedding dim与hidden dim大小实现了不同的训练效果

1）embedding dim = 300, hidden dim = 150（train\_set2）



2) embedding dim = 200, hidden dim = 100 (train\_set1)



(然后忘记截图了，大概与上述的差不多，在几个epoch后收敛于94%-95%间的准确率)

3）embedding dim = 7，hidden dim = 4

忘记划分验证集与训练集orzzzz，但是效果明显不好，大概在check函数上的正确率只有80%出头。由此可见，embedding dim和hidden dim的大小还是大一点效果会更好